ADVERSARIAL ADAPTIVE SAMPLING: UNIFY PINN AND OPTIMAL TRANSPORT FOR THE APPROXIMATION OF PDES

Anonymous authors
Paper under double-blind review

ABSTRACT

Solving partial differential equations (PDEs) is a central task in scientific computing. Recently, neural network approximation of PDEs has received increasing attention due to its flexible meshless discretization and its potential for high-dimensional problems. One fundamental numerical difficulty is that random samples in the training set introduce statistical errors into the discretization of the loss functional which may become the dominant error in the final approximation, and therefore overshadow the modeling capability of the neural network. In this work, we propose a new minmax formulation to optimize simultaneously the approximate solution, given by a neural network model, and the random samples in the training set, provided by a deep generative model. The key idea is to use a deep generative model to adjust the random samples in the training set such that the residual induced by the neural network model can maintain a smooth profile in the training process. Such an idea is achieved by implicitly embedding the Wasserstein distance between the residual-induced distribution and the uniform distribution into the loss, which is then minimized together with the residual. A nearly uniform residual profile means that its variance is small for any normalized weight function such that the Monte Carlo approximation error of the loss functional is reduced significantly for a certain sample size. The adversarial adaptive sampling (AAS) approach proposed in this work is the first attempt to formulate two essential components, minimizing the residual and seeking the optimal training set, into one minmax objective functional for the neural network approximation of PDEs.

1 INTRODUCTION

Partial differential equations (PDEs) are widely used to model physical phenomena. Typically, obtaining analytical solutions to PDEs is intractable, and thus numerical methods (e.g., finite element methods Elman et al. (2014)) have to be developed to approximate the solutions of PDEs. However, classical numerical methods can be computationally infeasible for high-dimensional PDEs due to the curse of dimensionality or computationally expensive for parametric low-dimensional PDEs Xiu & Karniadakis (2003); Ghosh et al. (2022); Yin et al. (2023); Zhai et al. (2022). To alleviate these difficulties, machine learning (ML) techniques, e.g., physics-informed neural networks (PINN) Raissi et al. (2019) and deep Ritz method E & Yu (2018), have been adapted to approximate PDEs as surrogate models and have received increasing attention Han et al. (2018); Zhu & Zabaras (2018); Zhu et al. (2019); Weinan (2021); Karniadakis et al. (2021). The basic idea of deep learning methods for approximating PDEs is to encode the information of PDEs in neural networks through a proper loss functional, which will be discretized by collocation points in the computational domain and subsequently minimized to determine an optimal model parameter Raissi et al. (2019); E & Yu (2018); Sirignano & Spiliopoulos (2018); Zhu et al. (2019).

The collocation points are crucial to effectively train neural networks for PDEs because they provide an approximation of the loss functional. In the community of computer vision or natural language processing, it is well known that the performance of ML models is highly dependent on the quality of data (i.e., the training set). Similarly, if the selected collocation points fail to yield an accurate approximation of the loss functional, it is not surprising that the trained neural network will suffer a large generalization error, especially when the solution has low regularity or the problem dimension

is large. As shown in Tang et al. (2023); Wu et al. (2023), if the collocation points in the training set are refined according to a proper error indicator, the accuracy can be dramatically improved. This is similar to classical adaptive methods such as the adaptive finite element method Morin et al. (2002); Mekchay & Nochetto (2005). In this work, we propose a new framework, called *adversarial adaptive sampling* (AAS), that simultaneously optimizes the loss functional and the training set to seek neural network approximation for PDEs through a minmax formulation. More specifically, we minimize the residual and meanwhile push the residual-induced distribution to a uniform distribution. To do this, we introduce a deep generative model into the AAS formulation, which not only provides random samples for the discretization of the loss functional, but also plays the role of the critic in WGAN Arjovsky et al. (2017); Gulrajani et al. (2017). In the maximization step, the deep generative model helps identify the difference in a Wasserstein distance between the residual-induced distribution and a uniform distribution; in the minimization step, such a difference is minimized together with the residual. This way, variance reduction is achieved once the residual profile is smoothed and the loss functional can be better approximated by a fixed number of random samples, which yields a more effective optimal model parameter, i.e., a more accurate neural network approximation of the PDE solution.

The main contributions of this paper are as follows.

- We unify PINN and optimal transport into an adversarial adaptive sampling framework, which provides a new perspective on neural network methods for solving PDEs.
- We develop a theoretical understanding of AAS and propose a simple but effective algorithm.

2 PINN AND ITS STATISTICAL ERRORS

The PDE problem considered here is: find $u \in \mathscr{F} : \Omega \mapsto \mathbb{R}$ where $\mathscr{F}$ is a proper function space defined on a computational domain $\Omega \in \mathbb{R}^D$, such that

$$\begin{aligned} \mathcal{L}u(\boldsymbol{x}) &= s(\boldsymbol{x}), \quad \forall \boldsymbol{x} \in \Omega \\ \mathfrak{b}u(\boldsymbol{x}) &= g(\boldsymbol{x}), \quad \forall \boldsymbol{x} \in \partial\Omega, \end{aligned} \tag{1}$$

where $\mathcal{L}$ is a partial differential operator (e.g., the Laplace operator Δ), $\mathfrak{b}$ is a boundary operator (e.g., the Dirichlet boundary), s is the source function, and g represents the boundary conditions. In the framework of PINN Raissi et al. (2019), the solution u of equation equation 1 is approximated by a neural network $u_{\boldsymbol{\theta}}(\boldsymbol{x})$ (parameterized with $\boldsymbol{\theta}$). The parameters $\boldsymbol{\theta}$ is determined by minimizing the following loss functional

$$\begin{aligned} &J\left(u_{\boldsymbol{\theta}}\right) = J_r(u_{\boldsymbol{\theta}}) + \gamma J_b(u_{\boldsymbol{\theta}}) \quad \text{with} \\ &J_r(u_{\boldsymbol{\theta}}) = \int_{\Omega} |r(\boldsymbol{x};\boldsymbol{\theta})|^2 d\boldsymbol{x} \ \text{ and } \ J_b(u_{\boldsymbol{\theta}}) = \int_{\partial\Omega} |b(\boldsymbol{x};\boldsymbol{\theta})|^2 d\boldsymbol{x}, \end{aligned} \tag{2}$$

where $r(\boldsymbol{x};\boldsymbol{\theta}) = \mathcal{L}u_{\boldsymbol{\theta}}(\boldsymbol{x}) - s(\boldsymbol{x})$, and $b(\boldsymbol{x};\boldsymbol{\theta}) = \mathfrak{b}u_{\boldsymbol{\theta}}(\boldsymbol{x}) - g(\boldsymbol{x})$ are the residuals that measure how well $u_{\boldsymbol{\theta}}$ satisfies the partial differential equations and the boundary conditions, respectively, and $\gamma > 0$ is a penalty parameter. To optimize this loss functional with respect to $\boldsymbol{\theta}$, we need to discretize the integral defined in equation 2 numerically. Let $\mathsf{S}_{\Omega} = \{\boldsymbol{x}_{\Omega}^{(i)}\}_{i=1}^{N_r}$ and $\mathsf{S}_{\partial\Omega} = \{\boldsymbol{x}_{\partial\Omega}^{(i)}\}_{i=1}^{N_b}$ be two sets of uniformly distributed collocation points on Ω and $\partial\Omega$ respectively. We then minimize the following empirical loss in practice

$$J_N\left(u_{\boldsymbol{\theta}}\right) = \frac{1}{N_r}\sum_{i=1}^{N_r} r^2(\boldsymbol{x}_{\Omega}^{(i)};\boldsymbol{\theta}) + \gamma\frac{1}{N_b}\sum_{i=1}^{N_b} b^2(\boldsymbol{x}_{\partial\Omega}^{(i)};\boldsymbol{\theta}), \tag{3}$$

which can be regarded as the Monte Carlo (MC) approximation of $J(u_{\boldsymbol{\theta}})$ subject to a statistical error of $O(N^{-1/2})$ with N being the sample size. Let $u_{\boldsymbol{\theta}_N^*}$ be the minimizer of the empirical loss $J_N(u_{\boldsymbol{\theta}})$ and $u_{\boldsymbol{\theta}^*}$ be the minimizer of the original loss functional $J(u_{\boldsymbol{\theta}})$. We can decompose the error into two parts as follows

$$\mathbb{E}\left(\left\|u_{\boldsymbol{\theta}_N^*} - u\right\|_{\Omega}\right) \leq \mathbb{E}\left(\left\|u_{\boldsymbol{\theta}_N^*} - u_{\boldsymbol{\theta}^*}\right\|_{\Omega}\right) + \left\|u_{\boldsymbol{\theta}^*} - u\right\|_{\Omega},$$

where $\mathbb{E}$ denotes the expectation operator and the norm $\|\cdot\|_\Omega$ corresponds to the function space $\mathscr{F}$ for u. One can see that the total error of neural network approximation for PDEs comes from two main aspects: the approximation error and the statistical error. The approximation error is dependent on the model capability of neural networks, while the statistical error originates from the collocation points.

Uniformly distributed collocation points are not effective for training neural networks if the solution has low regularity Tang et al. (2022; 2023); Wu et al. (2023) since the effective sample size of the MC approximation of $J(u_{\boldsymbol{\theta}})$ is significantly reduced by the large variance induced by the low regularity. For high-dimensional problems, information becomes more sparse or localized due to the curse of dimensionality, which shares some similarities with the low-dimensional problems of low regularity. Adaptive sampling is needed. In this work, we propose a new framework to optimize both the approximation solution and the training set.

3 Adversarial Adaptive Sampling

Adversarial adaptive sampling (AAS) includes two components to be optimized. One is a neural network $u_{\boldsymbol{\theta}}$ for approximating the PDE solution, and another is a probability density function (PDF) model $p_{\boldsymbol{\alpha}}$ (parameterized with $\boldsymbol{\alpha}$) for sampling. Unlike the deep adaptive sampling method (DAS) presented in Tang et al. (2023), in AAS, we simultaneously optimize the two models through an adversarial training procedure, which provides a new perspective to understand the role of random samples for the neural network approximation of PDEs.

3.1 A minmax formulation

The adversarial adaptive sampling approach can be formulated as the following minmax problem

$$\min_{\boldsymbol{\theta}} \max_{p_{\boldsymbol{\alpha}} \in V} \mathcal{J}(u_{\boldsymbol{\theta}}, p_{\boldsymbol{\alpha}}) = \int_\Omega r^2(\boldsymbol{x};\boldsymbol{\theta}) p_{\boldsymbol{\alpha}}(\boldsymbol{x}) d\boldsymbol{x} + \gamma J_b(u_{\boldsymbol{\theta}}), \tag{4}$$

where V is a function space that defines a proper constraint on $p_{\boldsymbol{\alpha}}(\boldsymbol{x})$. The choice of V will be specified in sections 3.2 and 3.3 in terms of the theoretical understanding and numerical implementation of AAS.

The main difference between $\mathcal{J}(u_{\boldsymbol{\theta}}, p_{\boldsymbol{\alpha}})$ and $J(u_{\boldsymbol{\theta}})$ in equation equation 2 is that the weight function for the integration of $r^2(\boldsymbol{x};\boldsymbol{\theta})$ is relaxed to $p_{\boldsymbol{\alpha}}(\boldsymbol{x})$ from a uniform one. First of all, such a relaxation can also be applied to $J_b(\cdot)$. In this work, we focus on the integration of r^2 for simplicity and assume that $J_b(\cdot)$ is well approximated by a prescribed set $\mathsf{S}_{\partial\Omega}$. Indeed, some penalty-free techniques Berg & Nyström (2018); Sheng & Yang (2021) can be employed to remove $J_b(\cdot)$. Second, $p_{\boldsymbol{\alpha}}(\boldsymbol{x}) > 0$ is regarded as a PDF on Ω, and an extra constraint on $p_{\boldsymbol{\alpha}}$ is necessary. Otherwise, the maximization step will simply yield a delta measure, i.e.,

$$\delta(\boldsymbol{x} - \boldsymbol{x}_0) = \operatorname*{arg\,max}_{p>0, \int_\Omega p d\boldsymbol{x} = 1} \int_\Omega r^2(\boldsymbol{x};\boldsymbol{\theta}) p(\boldsymbol{x}) d\boldsymbol{x},$$

where $\boldsymbol{x}_0 = \arg\max_{\boldsymbol{x}\in\Omega} r^2(\boldsymbol{x};\boldsymbol{\theta})$. Nevertheless, the region of large residuals is of particular importance for adaptive sampling. Third, the maximization in terms of $p_{\boldsymbol{\alpha}}$ is important numerically rather than theoretically. Indeed, if the statistical error does not exist and the model $u_{\boldsymbol{\theta}}$ includes the exact PDE solution, the minimum of r^2 is always reached at 0 as long as $p_{\boldsymbol{\alpha}}$ is positive on Ω. To reduce the statistical error induced by the random samples from $p_{\boldsymbol{\alpha}}$, we expect a small variance $\mathrm{Var}(r^2)$ in terms of $p_{\boldsymbol{\alpha}}$. If the variance of the Monte Carlo integration for r^2 is smaller than the variance of r^2 in terms of the uniform distribution, the accuracy of the Monte Carlo approximation will be improved for a fixed sample size, which yields a more accurate solution of PDEs Tang et al. (2023). To obtain a small $\mathrm{Var}(r^2)$, the profile of the residual needs to be nearly uniform. So an effective training strategy should not only minimize the residual but also endeavor to maintain a smooth profile of the residual, in other words, the two models $u_{\boldsymbol{\theta}}$ and $p_{\boldsymbol{\alpha}}$ need to work together. See Figure 1 for an informal description of the approach.

We will model $p_{\boldsymbol{\alpha}}$ using a bounded KRnet, which defines an invertible mapping $f_{\boldsymbol{\alpha}}(\cdot) : I^D \to I^D$ with $I = [-1, 1]$ and yields a normalizing flow model. The bounded KRnet can be achieved by

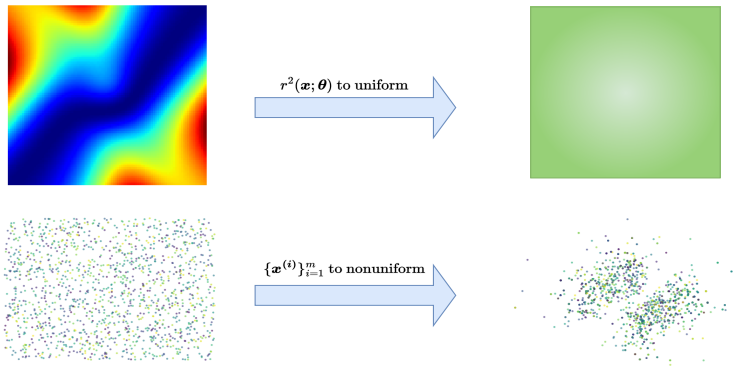


Figure 1: Two neural network models are simultaneously trained in the adversarial adaptive sampling framework. The residual is minimized and finally becomes "uniform", while the collocation points are updated and finally become nonuniform.

adding a logistic transformation layer Tang et al. (2023) or a new coupling layer proposed in Zeng et al. (2023). Let $\boldsymbol{z} = f_{\boldsymbol{\alpha}}(\boldsymbol{x})$ and $p_{\boldsymbol{Z}}(\boldsymbol{z})$ be a prior PDF. We define $p_{\boldsymbol{\alpha}}$ as

$$p_{\boldsymbol{\alpha}}(\boldsymbol{x}) = p_{\boldsymbol{Z}}(f_{\boldsymbol{\alpha}}(\boldsymbol{x}))|\nabla_{\boldsymbol{x}} f_{\boldsymbol{\alpha}}|.$$

Depending on a priori knowledge of the problem, the prior $p_{\boldsymbol{Z}}(\boldsymbol{z})$ can be chosen as a uniform distribution or more general models such as Gaussian mixture model.

3.2 Understanding of AAS

For simplicity and clarity, we remove $J_b(u_{\boldsymbol{\theta}})$ and consider

$$\min_{\boldsymbol{\theta}} \max_{p\in V} \mathcal{J}(u_{\boldsymbol{\theta}}, p) = \int_\Omega r^2(\boldsymbol{x};\boldsymbol{\theta}) p(\boldsymbol{x}) d\boldsymbol{x}. \tag{5}$$

We choose V as

$$V := \{p(\boldsymbol{x}) | \|p\|_{\text{Lip}} \le 1,\ 0 \le p(\boldsymbol{x}) \le M\},$$

where M is a positive number. We define a bounded metric

$$d_M(\boldsymbol{x},\boldsymbol{y}) = \min\{M, d(\boldsymbol{x},\boldsymbol{y})\}, \quad \boldsymbol{x},\boldsymbol{y} \in \mathbb{R}^D,$$

where $d(\boldsymbol{x},\boldsymbol{y}) = \|\boldsymbol{x}-\boldsymbol{y}\|_2$ is the Euclidean metric in $\mathbb{R}^D$. Without loss of generality, let Ω be a compact subset of $\mathbb{R}^D$ with total Lebesgue measure 1, and μ and ν two probability measures on Ω. The *Wasserstein distance* $d_{W^M}(\mu,\nu)$ between μ and ν for the metric $d_M(\boldsymbol{x},\boldsymbol{y})$ is

$$d_{W^M}(\mu,\nu) = \inf_{\pi\in\Pi(\Omega\times\Omega)} \int_{\Omega\times\Omega} d_M(\boldsymbol{x},\boldsymbol{y})\, d\pi(\boldsymbol{x},\boldsymbol{y}),$$

where $\Pi(\Omega\times\Omega)$ is the collection of all joint probability measures on $\Omega\times\Omega$. The dual form (see e.g. Villani (2003), Theorem 1.14 and Remark 1.15 on Page 34) of d_{W^M} is

$$d_{W^M}(\mu,\nu) = \sup\left\{ \int_\Omega \phi(\boldsymbol{x})\, d(\mu-\nu)(\boldsymbol{x}) \middle| \, 0 \le \phi(\boldsymbol{x}) \le \|d_M\|_\infty = M, \text{ and } \|\phi\|_{\text{Lip}} \le 1 \right\}, \tag{6}$$

where $\|\phi\|_{\text{Lip}}$ is the Lipschitz norm of function ϕ. We now reformulate the maximization problem as

$$\begin{aligned}
&\sup_{p\in V} \int_\Omega r^2(\boldsymbol{x};\theta) p(\boldsymbol{x}) d\boldsymbol{x} \\
=&\sup_{p\in V} \int_\Omega r^2(\boldsymbol{x};\theta) p(\boldsymbol{x}) d\boldsymbol{x} - \int_\Omega r^2(\boldsymbol{x};\theta) d\boldsymbol{x} \int_\Omega p(\boldsymbol{x}) d\boldsymbol{x} + \int_\Omega r^2(\boldsymbol{x};\theta) d\boldsymbol{x} \int_\Omega p(\boldsymbol{x}) d\boldsymbol{x} \\
\le& \int_\Omega r^2(\boldsymbol{x};\theta) d\boldsymbol{x} \left(\sup_{p\in V} \left[\int_\Omega p(\boldsymbol{x}) d\mu_r - \int_\Omega p(\boldsymbol{x}) d\mu_u \right] + \sup_{p\in V} \int_\Omega p(\boldsymbol{x}) d\boldsymbol{x} \right) \\
\le& (d_{W^M}(\mu_r,\mu_u) + M) \int_\Omega r^2(\boldsymbol{x};\boldsymbol{\theta})\, d\boldsymbol{x},
\end{aligned}$$

where μ_r and μ_u indicate the probability measures on Ω induced by $r^2(\boldsymbol{x})$ and the uniform distribution on Ω respectively. It can be shown that the constant M exists if we modify the function space V as

$$\hat{V} = \{p(\boldsymbol{x}) | \|p\|_{\text{Lip}} \leq 1,\, p(\boldsymbol{x}) \geq 0, \int_\Omega p(\boldsymbol{x}) d\boldsymbol{x} = 1\},$$

where $p(\boldsymbol{x})$ can then be regarded as a PDF. It is seen that the upper bound includes both the loss of the standard PINN and the Wasserstein distance between the residual induced distribution μ_r and the uniform distribution μ_u. For any u, the existence of function $\tilde{u}$, which has the same total residual loss as u and a uniform residual profile, is theoretically guaranteed (for detailed construction and these properties of $\tilde{u}$, please see the proof of Theorem 4 in Appendix A.2). This ensures that we can simultaneously reduce the residual and the Wasserstein distance between the (renormalized) residual and the uniform distribution. Once the residual profile is smoothed, variance reduction is achieved such that the Monte Carlo approximation of $\mathcal{J}(u_{\boldsymbol{\theta}}, p)$ will be more accurate for a fixed sample size. This eventually reduces the statistical error of the approximate PDE solution.

We now summarize our main analytical results. Consider

$$\inf_u \sup_{p \in \hat{V}} \mathcal{J}(u, p) = \int_\Omega r^2(u(\boldsymbol{x})) p(\boldsymbol{x})\, d\boldsymbol{x}, \tag{7}$$

with the following assumption.

Assumption A1. The operator r in equation 7 is a surjection from a function space $E_1(\mathbb{R}^D)$ to $C_c^\infty(\Omega)$, the class of C^∞ functions that are compactly supported on Ω.

In general, $E_1(\mathbb{R}^D)$ can be any function space, such as space of neural networks, smooth functions, or Sobolev spaces. And this assumption means for any smooth function $f \in C_c^\infty(\Omega)$, equation $r^2(u^*) = f$ admits some solution u^*. For example, if r is Laplacian Δ, the assumption means we can find a solution for $\Delta u = f$ for any f in $C_c^\infty(\Omega)$. With this assumption, we can prove the following main theorem for the min-max problem equation 7 (for the detailed proof, please see the Section A.2 in the supplementary material),

Theorem 1. *Let μ be the Lebesgue measure on $\mathbb{R}^D$, which represents the uniform probability distribution on Ω. In addition, we assume Assumption A1 holds. Then the optimal value of the min-max problem equation 7 is 0. Moreover, there is a sequence $\{u_n\}_{n=1}^\infty$ of functions with $r(u_n) \neq 0$ for all n, such that it is an optimization sequence of equation 7, namely,*

$$\lim_{n \to \infty} \mathcal{J}(u_n, p_n) = 0,$$

for some sequence of functions $\{p_n\}_{n=1}^\infty$ satisfying the constraints in equation 7. Meanwhile, this optimization sequence has the following two properties:

1. *The residual sequence $\{r(u_n)\}_{n=1}^\infty$ of $\{u_n\}_{n=1}^\infty$ converges to 0 in $L^2(d\mu)$.*
2. *The renormalized squared residual distributions*

$$d\nu_n \triangleq \frac{r^2(u_n)}{\int_\Omega r^2(u_n(\boldsymbol{x}))\, d\boldsymbol{x}}\, d\mu(\boldsymbol{x})$$

converge to the uniform distribution μ in the Wasserstein distance d_{W^M}.

3.3 Implementation of AAS

In the previous section, we have shown that $p_{\boldsymbol{\alpha}}(\boldsymbol{x})$ and $u_{\boldsymbol{\theta}}(\boldsymbol{x})$ in equation equation 4 play a similar role as the critic and generator in WGAN Arjovsky et al. (2017); Gulrajani et al. (2017). The generator of WGAN minimizes the Wasserstein distance between two distributions; PINN minimizes the residual; AAS achieves a tradeoff between the minimization of the residual and the minimization of the Wasserstein distance between the residual-induced distribution and the uniform distribution. From the implementation point of view, a particular difficulty is the constraint $\|p\|_{\text{Lip}} \leq 1$ induced by the function space $\hat{V}$. In this work, we propose a weaker constraint that can be easily implemented. We consider

$$\min_{\boldsymbol{\theta}} \max_{\substack{p_{\boldsymbol{\alpha}} > 0, \\ \int_\Omega p_{\boldsymbol{\alpha}}(\boldsymbol{x}) d\boldsymbol{x} = 1}} \mathcal{J}(u_{\boldsymbol{\theta}}, p_{\boldsymbol{\alpha}}) = \int_\Omega r^2(\boldsymbol{x}; \boldsymbol{\theta}) p_{\boldsymbol{\alpha}}(\boldsymbol{x}) d\boldsymbol{x} - \beta \int_\Omega |\nabla_{\boldsymbol{x}} p_{\boldsymbol{\alpha}}(\boldsymbol{x})|^2 d\boldsymbol{x}, \tag{8}$$

where we use a H_1 regularization term to replace explicit control on the Lipschitz condition. The constraints on a PDF are naturally satisfied because $p_{\boldsymbol{\alpha}}$ is a normalizing flow model. $p_{\boldsymbol{\alpha}}(\boldsymbol{x}) > 0$ as long as the prior is positive since $f_{\boldsymbol{\alpha}}(\cdot)$ is an invertible mapping. It can be shown that the maximizer for a fixed $u_{\boldsymbol{\theta}}$ is uniquely determined by the following elliptic equation

$$\begin{cases} 2\beta\nabla^2 p^* + r^2(\boldsymbol{x};\boldsymbol{\theta}) - \frac{1}{|\Omega|}\int_\Omega r^2(\boldsymbol{x};\boldsymbol{\theta})d\boldsymbol{x} = 0, & \boldsymbol{x}\in\Omega, \\ \frac{\partial p^*}{\partial \boldsymbol{n}} = 0, & \boldsymbol{x}\in\partial\Omega. \end{cases} \tag{9}$$

In the deep learning framework, the neural networks are in general (particularly when solving PDEs) differentiable. So the regularity constraint $\|p^*\|_{\text{Lip}} \leq M$ is equivalent to $\|\nabla p^*\|_\infty \leq M$ on a compact set Ω. Thus, we can adjust the penalty parameter β to implicitly control this regularity. Such a choice is demonstrated to be empirically sufficient since we focus on PDE approximation instead of PDF approximation.

To update $\boldsymbol{\theta}$ at the minimization step, we approximate the first term of $\mathcal{J}(u_{\boldsymbol{\theta}}, p_{\boldsymbol{\alpha}})$ in equation 8 using Monte Carlo methods:

$$\int_\Omega r^2\left[u_{\boldsymbol{\theta}}(\boldsymbol{x})\right]p_{\boldsymbol{\alpha}}(\boldsymbol{x})d\boldsymbol{x} \approx \frac{1}{m}\sum_{i=1}^m r^2\left[u_{\boldsymbol{\theta}}(\boldsymbol{x}_{\boldsymbol{\alpha}}^{(i)})\right], \tag{10}$$

where $\boldsymbol{x}_{\boldsymbol{\alpha}}^{(i)}$ can be generated from the probability density $p_{\boldsymbol{\alpha}}$ efficiently thanks to the invertible mapping $f_{\boldsymbol{\alpha}}(\cdot)$. To update $\boldsymbol{\alpha}$ at the maximization step, we approximate $\mathcal{J}(u_{\boldsymbol{\theta}}, p_{\boldsymbol{\alpha}})$ by importance sampling:

$$\mathcal{J}(u_{\boldsymbol{\theta}}, p_{\boldsymbol{\alpha}}) \approx \frac{1}{m}\sum_{i=1}^m \frac{r^2\left[u_{\boldsymbol{\theta}}(\boldsymbol{x}_{\boldsymbol{\alpha}'}^{(i)})\right]p_{\boldsymbol{\alpha}}(\boldsymbol{x}_{\boldsymbol{\alpha}'}^{(i)})}{p_{\boldsymbol{\alpha}'}(\boldsymbol{x}_{\boldsymbol{\alpha}'}^{(i)})} - \beta\cdot\frac{1}{m}\sum_{i=1}^m \frac{|\nabla_{\boldsymbol{x}} p_{\boldsymbol{\alpha}}(\boldsymbol{x}_{\boldsymbol{\alpha}'}^{(i)})|^2}{p_{\boldsymbol{\alpha}'}(\boldsymbol{x}_{\boldsymbol{\alpha}'}^{(i)})}, \tag{11}$$

where $p_{\boldsymbol{\alpha}'}$ is a PDF model with known parameters $\boldsymbol{\alpha}'$ and each $x_{\boldsymbol{\alpha}'}^{(i)}$ is a sample drawn from $p_{\boldsymbol{\alpha}'}$. Using equation equation 10 and equation 11, we can compute the gradient with respect to $\boldsymbol{\theta}$ and $\boldsymbol{\alpha}$, and the parameters can be updated by gradient-based optimization methods (e.g., Adam Kingma & Ba (2017)). The training procedure is similar to GAN Goodfellow et al. (2014) and can be summarized in Algorithm 1, where we let $p_{\boldsymbol{\alpha}'} = p_{\boldsymbol{\alpha}_k}$ in equation equation 11, i.e., the PDF model from the last step is used for importance sampling when computing $\mathcal{J}(u_{\boldsymbol{\theta}}, p_{\boldsymbol{\alpha}})$.

Algorithm 1 AAS for PDEs

Input: Initial $p_{\boldsymbol{\alpha}}$ and $u_{\boldsymbol{\theta}}$, maximal iteration M, batch size m, initial training set $\mathsf{S}_{\Omega,0} = \{\boldsymbol{x}_{\boldsymbol{\alpha}_0}^{(i)}\}_{i=1}^{N_r}$ and $\mathsf{S}_{\partial\Omega,0} = \{\boldsymbol{x}_{\partial\Omega,0}^{(i)}\}_{i=1}^{N_b}$.

```
for k = 0, ..., M do
    for j steps do
        Sample m samples from S_{Ω,k}.
        Sample m samples from S_{∂Ω,k}.
        Update u_θ by descending the stochastic gradient of J(θ, α) (see equation equation 10).
    end for
    for j steps do
        Sample m samples from S_{Ω,k}.
        Update p_α by ascending the stochastic gradient of J(θ, α) (see equation equation 11).
    end for
    Generate S_{Ω,k+1} ⊂ Ω through p_{α_k}.
end for
```

Output: $u_{\boldsymbol{\theta}}$

4 Related Work

There is a lot of related work, and we summarize the most related lines of this work.

Adaptive sampling. Adaptive sampling methods have been receiving increasing attention in solving PDEs with deep learning methods. The basic idea of such methods is to define a proper error

indicator Wu et al. (2023); Yu et al. (2022) for refining collocation points in the training set, in which sampling approaches Gao & Wang (2023) (e.g., Markov Chain Monte Carlo) or deep generative models Tang et al. (2023) are often invoked. To this end, an additional deep generative model (e.g., normalizing flow models), or classical PDF model (e.g., Gaussian mixture models Gao et al. (2022); Jiao et al. (2023)) for sampling is usually required, which is similar to this work. However, there are some crucial differences between existing approaches and the proposed adversarial adaptive sampling (AAS) framework. First, in AAS, the evolution of the residual-induced distribution has a clear path. That is, this residual-induced distribution is pushed to a uniform distribution during training. Because minimizing the Wasserstein distance between the residual-induced distribution and the uniform distribution is naturally embedded in the loss functional in the proposed adversarial sampling framework. Second, unlike the existing methods, our AAS method admits an adversarial training style like in WGAN, which is the first time to minimize the residual and seek the optimal training set simultaneously for PINN.

Adversarial training. In Zang et al. (2020), the authors proposed a weak formulation with primal and adversarial networks, where the PDE problem is converted to an operator norm minimization problem derived from the weak formulation. Although the adversarial training procedure is employed in Zang et al. (2020), it does not involve the training set but the function space. Introducing one or more discriminator networks to construct adversarial training is studied in Zeng et al. (2022), where the discriminator is used for the reward that PINN predicts mistakes. However, this approach does not optimize the training set but implicitly assigns higher weights for samples with large pointwise residuals through adversarial training.

5 Numerical Results

We use some benchmark test problems presented in Tang et al. (2023) to demonstrate the proposed method. All models are trained by the Adam method Kingma & Ba (2017). The hyperparameters of neural networks are set to be the same as those in Tang et al. (2023). For comparison, we also implement the DAS algorithm Tang et al. (2023) and the RAR algorithm Lu et al. (2021); Yu et al. (2022) as the baseline models. The training of neural networks is performed on a Geforce RTX 3090 GPU with TensorFlow 2.0. The codes of all examples will be released on GitHub once the paper is accepted.

5.1 One-peak problem

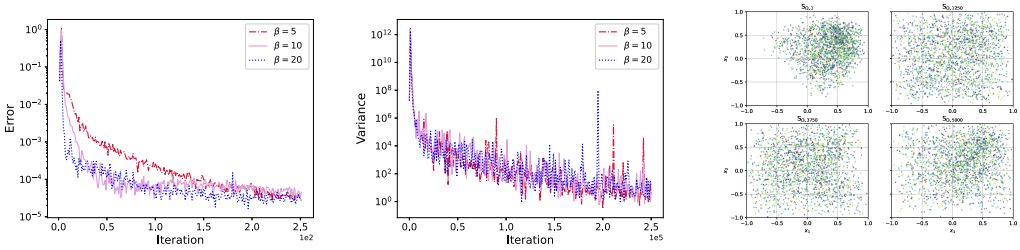
We start with the following equation which is a benchmark test problem for adaptive finite element methods Mitchell (2013); Morin et al. (2002):

$$\begin{aligned} -\Delta u(\boldsymbol{x}) &= s(\boldsymbol{x}) \quad \text{in } \Omega, \\ u(\boldsymbol{x}) &= g(\boldsymbol{x}) \quad \text{on } \partial\Omega, \end{aligned} \tag{12}$$

where $\boldsymbol{x} = [x_1, x_2]^\mathsf{T}$ and the computation domain is $\Omega = [-1, 1]^2$. The following reference solution is given by

$$u(x_1, x_2) = \exp\left(-1000[(x_1 - 0.5)^2 + (x_2 - 0.5)^2]\right),$$

which has a peak at $(0.5, 0.5)$ and decreases rapidly away from $(0.5, 0.5)$. The reference solution is imposed on the boundary. The source term $s(\boldsymbol{x})$ is derived by the exact solution and is listed in the supplementary A.5. A uniform meshgrid with size 256×256 in $[-1, 1]^2$ is generated and the error is defined to be the mean square error on these grid points. From Figure 2(a), it can be seen that our AAS method can give an accurate approximation for this peak test problem. Note that the uniform sampling strategy is not suitable for this test problem as studied in Tang et al. (2023). The training behaviour behavior for different regularization parameters (i.e., β) is shown in Figure 2(a) and Figure 2(b). It can be seen that the error behavior is similar for $\beta = 5, 10, 20$. Figure Figure 2(c) shows the evolution of the residual variance and the training set during training for $\beta = 5$, where the variance decreases as the training step increases and the training set finally concentrates on $(0.5, 0.5)$ with a heavy tail. The comparison of different adaptive sampling methods is presented in Table 1, which also included the resultd of the following test problems.



(a) (b) (c)

Figure 2: The results for the peak test problem. (a) The error behaviour. (b) The variance behavior. (c) The evolution of the training set.

5.2 TWO-PEAK PROBLEM

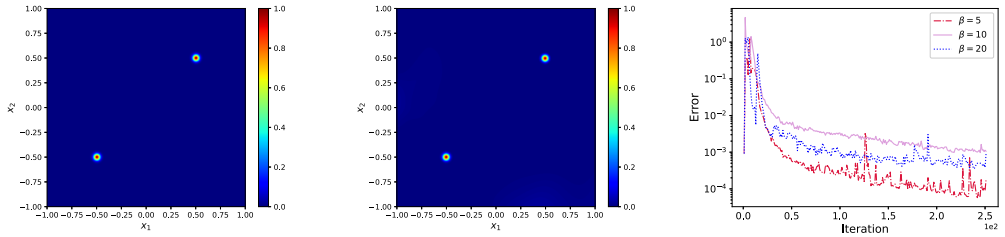
We next consider the following equation

$$\begin{aligned} -\nabla \cdot [u(\boldsymbol{x})\nabla v(\boldsymbol{x})] + \nabla^2 u(\boldsymbol{x}) &= s(\boldsymbol{x}) \quad \text{in } \Omega, \\ u(\boldsymbol{x}) &= g(\boldsymbol{x}) \quad \text{on } \partial\Omega, \end{aligned} \tag{13}$$

where $\boldsymbol{x} = [x_1, x_2]^\mathsf{T}$, $v(\boldsymbol{x}) = x_1^2 + x_2^2$, and the computation domain is $\Omega = [-1, 1]^2$. Following Tang et al. (2023), the exact solution of equation 13 is set to be as

$$u(x_1, x_2) = \mathrm{e}^{-1000[(x_1-0.5)^2+(x_2-0.5)^2]} + \mathrm{e}^{-1000[(x_1+0.5)^2+(x_2+0.5)^2]},$$

which has two peaks at the points $(0.5, 0.5)$ and $(-0.5, -0.5)$. Here, the Dirichlet boundary condition on $\partial\Omega$ is given by the exact solution. From Figure 3(a) and Figure 3(b), it can be seen that our AAS method can give an accurate approximation for this two-peak test problem. The error behavior for different regularization parameters (i.e., β) is shown in Figure 3(c). Figure 4 shows the evolution of the residual variance and the training set during training for $\beta = 5$, where the residual variance decreases as the training step increases and the training set finally concentrates on $(-0.5, -0.5)$and $(0.5, 0.5)$ with a heavy tail.



(a) (b) (c)

Figure 3: The results for the two-peak test problem. (a) The exact solution. (b) AAS approximation. (c) The error behavior.

5.3 HIGH-DIMENSIONAL NONLINEAR EQUATION

In this part, we consider the following ten-dimensional nonlinear partial differential equation

$$\begin{aligned} -\Delta u(\boldsymbol{x}) + u(\boldsymbol{x}) - u^3(\boldsymbol{x}) &= s(\boldsymbol{x}), \quad \boldsymbol{x} \text{ in } \Omega = [-1, 1]^{10} \\ u(\boldsymbol{x}) &= g(\boldsymbol{x}), \quad \boldsymbol{x} \text{ on } \partial\Omega. \end{aligned} \tag{14}$$

The exact solution is $u(\boldsymbol{x}) = \mathrm{e}^{-10\|\boldsymbol{x}\|_2^2}$ and the Dirichlet boundary condition on $\partial\Omega$ is imposed by the exact solution. The source term $s(\boldsymbol{x})$ is derived by the exact solution and is listed in the supplementary A.5. The error is defined to be the same as in Tang et al. (2023). Figure 5 shows the results

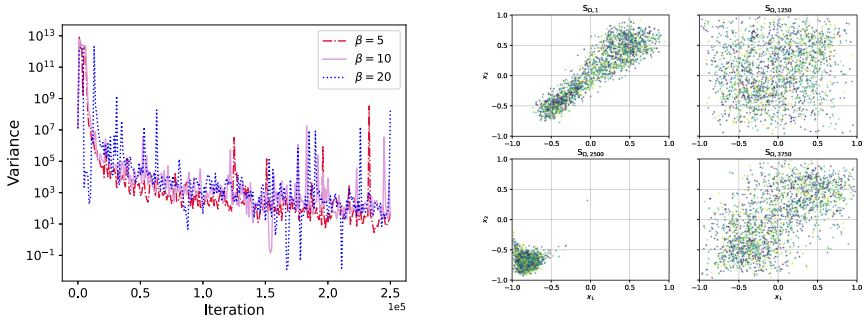
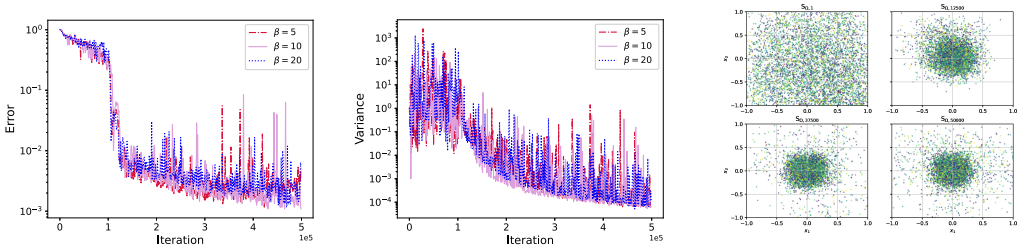


Figure 4: The evolution of the residual variance and the training set for the two-peak test problem. Left: The variance behavior. Right: The evolution of the training set.

of the ten-dimensional nonlinear test problem. Specifically, Figure 5(a) shows the error behavior during training for different regularization parameters, and Figure 5(b) shows the evolution of the residual variance. Figure 5(c) shows the samples during the adversarial training process, where we select the components x_1 and x_2 for visualization. We have also checked the other components, and the results are similar. It is seen that the training set finally becomes nonuniform to get a small residual variance. The results of different adaptive sampling strategies for the three test problems are summarized in Table 1.



(a) (b) (c)

Figure 5: The results of the ten-dimensional nonlinear test problem. (a) The error behavior. (b) The variance behaviour. (c) The evolution of the training set, $x_1 - x_2$ plane ($\beta = 10$).

Table 1: Error comparison of adaptive sampling methods

Method \ Test problem	PDE equation 12	PDE equation 13	PDE equation 14
PINN	9.74e-04	3.22e-02	1.01
RAR Lu et al. (2021)	-	-	9.83e-01
DAS-G Tang et al. (2023)	3.75e-04	1.51e-03	9.55e-03
DAS-R Tang et al. (2023)	1.93e-04	6.21e-03	1.26e-02
AAS (this work)	**2.97e-05**	**1.09e-04**	**1.31e-03**

6 Conclusions

We developed a novel adversarial adaptive sampling (AAS) approach that unifies PINN and optimal transport for neural network approximation of PDEs. With AAS, the evolution of the training set can be investigated in terms of the optimal transport theory, and numerical results have demonstrated the importance of random samples for training PINN more effectively.

REFERENCES

Robert A Adams and J F John Fournier. *Sobolev Spaces, 2nd Edition*. Elsevier, Amsterdam, 2003.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International Conference on Machine Learning*, pp. 214–223. PMLR, 2017.

Jens Berg and Kaj Nyström. A unified deep artificial neural network approach to partial differential equations in complex geometries. *Neurocomputing*, 317:28–41, 2018.

Weinan E and Bing Yu. The deep Ritz method: A deep learning-based numerical algorithm for solving variational problems. *Communications in Mathematics and Statistics*, 6(1):1–12, 2018.

Howard C Elman, David J Silvester, and Andrew J Wathen. *Finite elements and fast iterative solvers: With applications in incompressible fluid dynamics*. Oxford University Press, USA, 2014.

Lawrence C Evans. *Partial Differential Equations*. American Mathematical Soc., 2010.

Wenhan Gao and Chunmei Wang. Active learning based sampling for high-dimensional nonlinear partial differential equations. *Journal of Computational Physics*, 475:111848, 2023.

Zhiwei Gao, Liang Yan, and Tao Zhou. Failure-informed adaptive sampling for PINNs. *arXiv preprint arXiv:2210.00279*, 2022.

Sayan Ghosh, Govinda Anantha Padmanabha, Cheng Peng, Valeria Andreoli, Steven Atkinson, Piyush Pandita, Thomas Vandeputte, Nicholas Zabaras, and Liping Wang. Inverse aerodynamic design of gas turbine blades using probabilistic machine learning. *Journal of Mechanical Design*, 144(2), 2022.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pp. 2672–2680, 2014.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of Wasserstein GANs. *Advances in Neural Information Processing Systems*, 30, 2017.

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

Yuling Jiao, Di Li, Xiliang Lu, Jerry Zhijian Yang, and Cheng Yuan. GAS: A Gaussian mixture distribution-based adaptive sampling method for PINNs. *arXiv preprint arXiv:2303.15849*, 2023.

George Em Karniadakis, Ioannis G Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2017.

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, 2021.

Khamron Mekchay and Ricardo H Nochetto. Convergence of adaptive finite element methods for general second order linear elliptic PDEs. *SIAM Journal on Numerical Analysis*, 43(5):1803–1827, 2005.

William F Mitchell. A collection of 2D elliptic problems for testing adaptive grid refinement algorithms. *Applied Mathematics and Computation*, 220:350–364, 2013.

Pedro Morin, Ricardo H Nochetto, and Kunibert G Siebert. Convergence of adaptive finite element methods. *SIAM Review*, 44(4):631–658, 2002.

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.

Hailong Sheng and Chao Yang. PFNN: A penalty-free neural network method for solving a class of second-order boundary-value problems on complex geometries. *Journal of Computational Physics*, 428:110085, 2021.

Justin Sirignano and Konstantinos Spiliopoulos. DGM: A deep learning algorithm for solving partial differential equations. *Journal of Computational Physics*, 375:1339–1364, 2018.

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.

Michael E Taylor. *Partial Differential Equations I: Basic Theory, 2nd Edition*. Springer, 2011.

Cédric Villani. *Topics in Optimal Transportation*. Number 58 in Graduate Studies in Mathematics. American Mathematical Society, 2003.

E Weinan. The dawning of a new era in applied mathematics. *Notices of the American Mathematical Society*, 68(4):565–571, 2021.

Chenxi Wu, Min Zhu, Qinyang Tan, Yadhu Kartha, and Lu Lu. A comprehensive study of non-adaptive and residual-based adaptive sampling for physics-informed neural networks. *Computer Methods in Applied Mechanics and Engineering*, 403:115671, 2023.

Dongbin Xiu and George Em Karniadakis. Modeling uncertainty in flow simulations via generalized polynomial chaos. *Journal of Computational Physics*, 187(1):137–167, 2003.

Pengfei Yin, Guangqiang Xiao, Kejun Tang, and Chao Yang. AONN: An adjoint-oriented neural network method for all-at-once solutions of parametric optimal control problems. *arXiv preprint arXiv:2302.02076*, 2023.

Jeremy Yu, Lu Lu, Xuhui Meng, and George Em Karniadakis. Gradient-enhanced physics-informed neural networks for forward and inverse PDE problems. *Computer Methods in Applied Mechanics and Engineering*, 393:114823, 2022.

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.

Li Zeng, Xiaoliang Wan, and Tao Zhou. Bounded krnet and its applications to density estimation and approximation. *arXiv:2305.09063*, 2023.

Qi Zeng, Spencer H Bryngelson, and Florian Tobias Schaefer. Competitive physics informed networks. In *ICLR 2022 Workshop on Gamification and Multiagent Solutions*, 2022.

Jiayu Zhai, Matthew Dobson, and Yao Li. A deep learning method for solving Fokker-Planck equations. In *Mathematical and Scientific Machine Learning*, pp. 568–597. PMLR, 2022.

Yinhao Zhu and Nicholas Zabaras. Bayesian deep convolutional encoder–decoder networks for surrogate modeling and uncertainty quantification. *Journal of Computational Physics*, 366:415–447, 2018.

Yinhao Zhu, Nicholas Zabaras, Phaedon-Stelios Koutsourelakis, and Paris Perdikaris. Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data. *Journal of Computational Physics*, 394:56–81, 2019.