Learning from Integral Losses in Physics Informed Neural Networks

Ehsan Saleh[1] Saba Ghaffari[1] Timothy Bretl[2] Luke Olson[1] Matthew West[3]

Abstract

This work proposes a solution for the problem of training physics-informed networks under partial integro-differential equations. These equations require an infinite or a large number of neural evaluations to construct a single residual for training. As a result, accurate evaluation may be impractical, and we show that naive approximations at replacing these integrals with unbiased estimates lead to biased loss functions and solutions. To overcome this bias, we investigate three types of potential solutions: the deterministic sampling approaches, the double-sampling trick, and the delayed target method. We consider three classes of PDEs for benchmarking; one defining Poisson problems with singular charges and weak solutions of up to 10 dimensions, another involving weak solutions on electro-magnetic fields and a Maxwell equation, and a third one defining a Smoluchowski coagulation problem. Our numerical results confirm the existence of the aforementioned bias in practice and also show that our proposed delayed target approach can lead to accurate solutions with comparable quality to ones estimated with a large sample size integral. Our implementation is open-source and available at https://github.com/ehsansaleh/btspinn.

1. Introduction

Physics Informed Neural Networks (PINNs) (Raissi et al., 2019) can be described as solvers of a particular Partial Differential Equation (PDE). Typically, these problems consist of three defining elements. A sampling procedure selects a number of points for learning. Automatic differentiation is then used to evaluate the PDE at these points and define a residual. Finally, a loss function, such as the Mean Squared Error (MSE), is applied to these residuals, and the network learns the true solution by minimizing this loss through back-propagation and stochastic approximation. These elements form the basis of many methods capable of learning high-dimensional parameters. A wealth of existing work demonstrated the utility of this approach to solving a wide array of applications and PDE forms (Li et al., 2020; Shukla et al., 2021; Li et al., 2019).

One particular problem in this area is the prevalent assumption around our ability to accurately evaluate the PDE residuals for learning. In particular, partial integro-differential forms include integrals or large summations within them. These forms appear in a broad range of scientific applications including quantum physics (Laskin, 2000), aerosol modeling (Wang et al., 2022a), and ecology (Humphries et al., 2010). In such instances, an accurate evaluation of the PDE elements, even at a single point, can become impractical. Naive approximations, such as replacing integrals with unbiased estimates, can result in biased solutions, as we will show later. This work is dedicated to the problem of learning PINNs with loss functions containing a parametrized integral or summation.

One natural approach for learning PINNs with integral forms would be to use techniques such as importance sampling, numerical quadrature, or Quasi Monte Carlo (QMC) to estimate the integrals more accurately than a standard i.i.d. sampling approach. This follows the classical theory and such approaches have been investigated thoroughly in prior work (Caflisch, 1998; Evans & Swartz, 1995).

In this work, we consider an alternative approach, which we will show can be more effective than reducing the variance of the integral estimation. The methods we investigate are based around the idea of reducing the bias and the variance in the parameter gradients so that we can train effectively even if our loss functions are not accurately estimated. We consider three potential approaches to do this; the deterministic sampling approach, the double-sampling trick, and the delayed target method. As we will see, the delayed target approach, which is based upon ideas from learning temporal differences (Sutton, 1984; Mnih et al., 2015; Fujimoto et al., 2018), gives the best results, performing comparable or slightly better than accurate integral estimators (i.e., with $N = 100$ samples) using just a single sample ($N = 1$).

[1]Department of Computer Science [2]Department of Aerospace Engineering [3]Department of Mechanical Science and Engineering. University of Illinois Urbana-Champaign. Correspondence to: Ehsan Saleh <ehsans2@illinois.edu>.

Proceedings of the 41^{st} International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

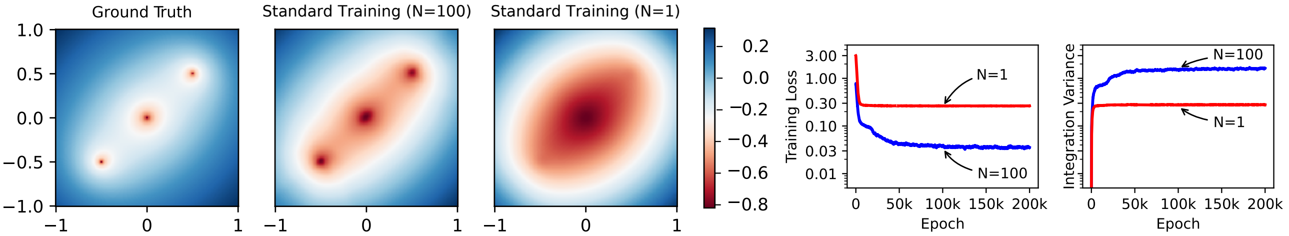

Figure 1. Training with the MSE loss under different sample sizes per surface (N). The heatmaps show the analytical solution (left), the low-variance training with $N = 100$ (middle), and the high-variance training with $N = 1$ (right). The smaller the N, the more biased the training objective becomes towards finding smoother solutions. The right panel shows the training curves; the training loss and the integration variance represent $\hat{\mathcal{L}}_\theta(x)$ and $\mathbb{V}_{P(x'|x)}[g_\theta(x')]$ in Equation (26), respectively. For $N = 1$, the training loss seems to be floored at the same value as the integration variance (i.e., approximately 0.3). However, with $N = 100$, the model produces better solutions, lower training losses, and higher integration variances.

Combining importance sampling and QMC methods with our techniques is a promising direction that we leave for future work.

The main contributions of this work are: (1) we formulate the integral learning problem under a general framework and show the biased nature of standard approximated loss functions; (2) we present three techniques to solve such problems, namely the deterministic sampling approaches, the double-sampling trick, and the delayed target method; (3) we detail an effective way of implementation for the delayed target method compared to a naive one; (4) we compare the efficacy of the potential solutions using numerical examples on Poisson problems with singular charges and up to 10 dimensions, a Maxwell problem with magnetic fields, and a Smoluchowski coagulation problem; (5) provide a convergence guarantee, approximation error upper bound, and computational complexity analysis for the delayed target method under linear function approximation.

2. Problem Formulation

Consider a typical partial integro-differential equation

$$f_\theta(x) := \mathbb{E}_{P(x'|x)}[g_\theta(x')] + y(x). \tag{1}$$

The $f_\theta(x)$ and $g_\theta(x')$ are parametrized, and $y(x)$ includes all the non-parametrized terms in the PDE. The right side of the equation serves as the target value for $f_\theta(x)$ (see Section A of the supplementary material for all the notation). Equation (1) is a general, yet concise, form for expressing partial integro-differential equations. To motivate this, we will express three examples in this form.

Example 2.1. The Poisson problem is to solve the system $\nabla^2 U = \rho$ for U given a charge function ρ. This is equivalent to finding a solution for a gradient and divergence system:

$$E = \nabla U, \tag{2}$$

$$\rho = \nabla \cdot E. \tag{3}$$

A weak solution can be obtained by enforcing the divergence theorem over many volumes:

$$\int_{\partial\Omega} E \cdot \hat{n} \quad \mathrm{d}S = \iint_{\Omega} \nabla \cdot E \quad \mathrm{d}V, \tag{4}$$

where $\hat{n}$ is the normal vector perpendicular to $\mathrm{d}S$. The weak solutions can be preferable over the strong ones when dealing with singular or sparse ρ charges.

To solve this system, we parametrize E as the gradient of a neural network predicting the U potentials. To convert this into the form of Equation (1), we replace the left integral in Equation (4) with an arbitrarily large Riemann sum as

$$\int_{\partial\Omega} E \cdot \hat{n} \, \mathrm{d}S = \frac{A}{M} \sum_{i=1}^{M} E_\theta(x_i) \cdot \hat{n}_i, \tag{5}$$

where $A = \int_{\partial\Omega} 1 \, \mathrm{d}S$ is the surface area and the x_i samples are uniform on the surface. To convert this system into the form of Equation (1), we define the following elements:

$$x := x_1, \tag{6}$$

$$f_\theta(x) := \frac{A}{M} E_\theta(x) \cdot \hat{n}_1, \tag{7}$$

$$g_\theta(x_i) := -\frac{A(M-1)}{M} E_\theta(x_i) \cdot \hat{n}_i, \tag{8}$$

$$P(x'|x_1) := \text{Unif}(\{x_2, \cdots, x_M\}), \tag{9}$$

$$y(x_1) := \iint_{\Omega} \rho \, \mathrm{d}V. \tag{10}$$

Example 2.2. In static electromagnetic conditions, one of the Maxwell Equations, the Ampere circuital law, is to solve the $\nabla \times A = B$ and $\nabla \times B = J$ system for A given the current density J in the 3D space (we assumed a unit physical coefficient for simplicity). Here, B represents the magnetic field and A denotes the magnetic potential vector. A weak solution for this system can be obtained by enforcing the Stokes theorem over many volumes:

$$\int_{\partial\Omega} \nabla \times A \cdot \mathrm{d}l = \iint_{\Omega} J \cdot \mathrm{d}S, \tag{11}$$

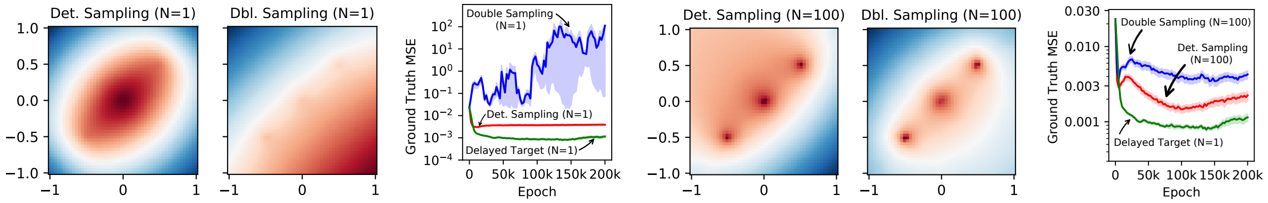

Figure 2. The results of the deterministic and double sampling techniques on the Poisson problem. The left plots demonstrate the solutions with $N = 1$, while the right plots show the solutions with $N = 100$. The training curves represent the mean squared error to the analytical solution vs. the training epochs. With $N = 1$, the double sampling trick exhibits divergence in training, and the deterministic sampling process yields overly smooth functions similar to the standard solution in Figure 1. However, with $N = 100$, both the deterministic and double-sampling approaches exhibit improvements. According to the training curves, the delayed target method with $N = 1$ yields the best solutions to this problem.

where $\mathrm{d}l$ and $\mathrm{d}S$ are infinitesimal surface tangent and normal vectors, respectively. Just like the Poisson problem, the weak solutions can be preferable over the strong ones when dealing with singular inputs, and this equation can be converted into the form of Equation (1) similarly.

Example 2.3. The Smoluchowski coagulation equation simulates the evolution of particles into larger ones and is described as

$$\frac{\partial n(x,t)}{\partial t} = \int_0^x K(x-x',x')n(x-x',t)n(x',t)\mathrm{d}x' - \int_0^\infty K(x,x')n(x,t)n(x',t)\mathrm{d}x', \tag{12}$$

where $K(x,x')$ is the coagulation kernel between two particles of size x and x'. The particle sizes x and x' can be generalized into vectors, inducing a higher-dimensional PDE to solve. To solve this problem, we parametrize $n(x,t)$ as the output of a neural model and write

$$f_\theta(x) := \frac{\partial n_\theta(x,t)}{\partial t}, \tag{13}$$

$$g_\theta^{(1)}(x') := A_1 K(x-x',x')n_\theta(x-x',t)n_\theta(x',t), \tag{14}$$

$$g_\theta^{(2)}(x') := A_2 K(x,x')n_\theta(x,t)n_\theta(x',t). \tag{15}$$

The x' values in both $g_\theta^{(1)}$ and $g_\theta^{(2)}$ are sampled from their respective uniform distributions, and A_1 and A_2 are used to normalize the uniform integrals into expectations. Finally, $y(x) := 0$ and we can define $g_\theta(x')$ in a way such that

$$\mathbb{E}_{x'}[g_\theta(x')] := \mathbb{E}_{x'}[g_\theta^{(1)}(x')] + \mathbb{E}_{x'}[g_\theta^{(2)}(x')]. \tag{16}$$

The standard way to solve systems such as Examples 2.1, 2.2, and 2.3 with PINNs, is to minimize the following mean squared error (MSE) loss (Raissi et al., 2019; Jagtap et al., 2020):

$$\mathcal{L}_\theta(x) := \big(f_\theta(x) - \mathbb{E}_{P(x'|x)}[g_\theta(x')] - y(x)\big)^2. \tag{17}$$

Algorithm 1 The regularized delayed target method

```
Require: The initial parameter values $\theta_0$, learning rate $\eta$, Polyak averaging rate $\tau$, target sample size $N$, and the target regularization weight $\lambda$.
Initialize the main and target parameters:$\theta, \theta_{\text{Target}} \leftarrow \theta_0$.
for $k = 1, 2, \ldots$ do
    Sample $x$ from $P$ and obtain the $y(x)$ label.
    Compute the $f_\theta(x)$ term using the main parameters.
    Obtain the $x'_1, \cdots, x'_N$ i.i.d. samples from $P(x'|x)$.
    Compute the $\frac{1}{N}\sum_{i=1}^N g_{\theta^{\text{Target}}}(x'_i) + y(x)$ target using the $\theta_{\text{Target}}$ target parameters.
    Construct the main loss:
        $\hat{\mathcal{L}}_\theta^{\text{DT}} = \big(f_\theta(x) - \frac{1}{N}\sum_{i=1}^N g_{\theta^{\text{Target}}}(x'_i) - y(x)\big)^2.$   (18)
    Construct the target regularization loss:
        $\hat{\mathcal{L}}_\theta^{\text{R}} = (f_\theta(x) - f_{\theta^{\text{Target}}}(x))^2.$   (19)
    Compute the total loss $\hat{\mathcal{L}}_\theta^{\text{DTR}} = \hat{\mathcal{L}}_\theta^{\text{DT}}(x) + \lambda\hat{\mathcal{L}}_\theta^{\text{R}}$.
    Perform a gradient descent step on $\theta$:
        $\theta \leftarrow \theta - \eta\nabla_\theta\hat{\mathcal{L}}_\theta^{\text{DTR}}.$   (20)
    Update the target parameters using Polyak averaging:
        $\theta^{\text{Target}} \leftarrow \tau\theta^{\text{Target}} + (1-\tau)\theta.$   (21)
```

Since computing exact integrals may be impractical, one may contemplate replacing the expectation in Equation (17) with an unbiased estimate, as implemented in NVIDIA's Modulus package (NVIDIA, 2024). This prompts the following approximate objective:

$$\hat{\mathcal{L}}_\theta(x) := \mathbb{E}_{P(x'_{1:n}|x)}\Big[\Big(f_\theta(x) - \frac{1}{N}\sum_{i=1}^N g_\theta(x'_i) - y(x)\Big)^2\Big]. \tag{22}$$

We therefore analyze the approximation error by adding and subtracting $\mathbb{E}_{x''}[g_\theta(x'')]$:

$$\hat{\mathcal{L}}_\theta(x) = \mathbb{E}_{P(x'_{1:n}|x)}\Big[\Big(\big(f_\theta(x) - \mathbb{E}_{x''}[g_\theta(x'')] - y(x)\big) + \big(\mathbb{E}_{x''}[g_\theta(x'')] - \frac{1}{N}\sum_{i=1}^N g_\theta(x'_i)\big)\Big)^2\Big]. \tag{23}$$

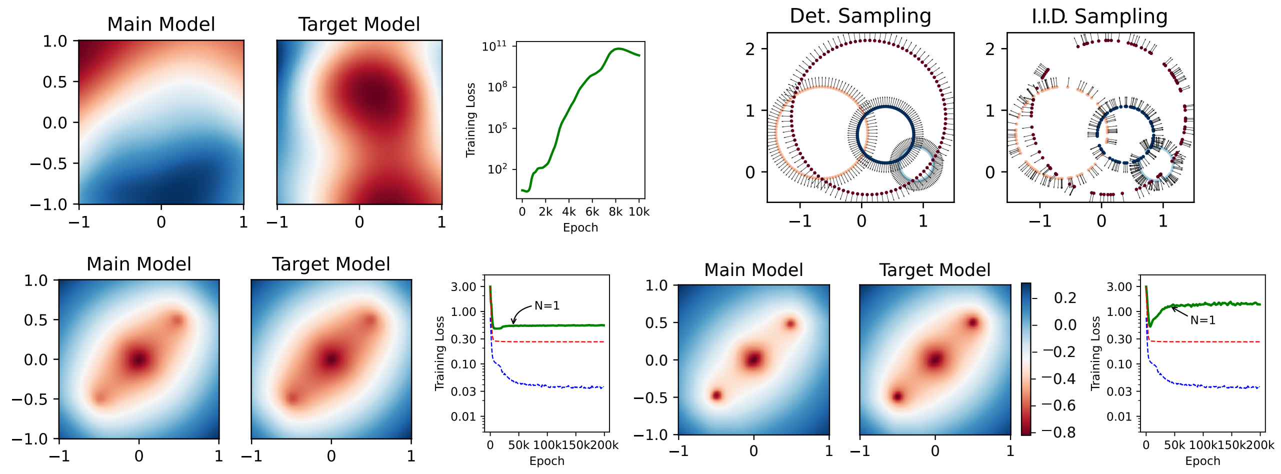

Figure 3. Training the same problem as in Figure 1 with delayed targets and $N = 1$. *The top left panel* shows a diverged training with $M = 100$ in Equation (37). *The lower left panel* corresponds to $M = 10$, which has a converging training curve even though it produces an overly smooth solution. In *the lower right panel*, we set $\lambda = 1$ which allowed setting $M = 1000$ while maintaining a stable training loss. In each panel, the left and right heatmaps show the main and the target model predictions, respectively, and the right plots show the training curves. The green curves show the training loss for the delayed target method, and the standard training curves with $N = 1$ and 100 are also shown using dotted red and blue lines for comparison, respectively. *The top right panel* shows an example of deterministic vs. i.i.d. sampling of the surface points in the Poisson problem. For each sampled sphere, the surface points and their normal vectors are shown with $N = 100$ samples. With deterministic sampling, the points are evenly spaced to cover the sampling domain.

By decomposing the squared sum, we get

$$\begin{aligned}\hat{\mathcal{L}}_\theta(x) = &\ \big(f_\theta(x) - \mathbb{E}_{x''}[g_\theta(x'')] - y(x)\big)^2 + \\ &\mathbb{E}_{P(x'_{1:n}|x)}\Big[\Big(\mathbb{E}_{x''}[g_\theta(x'')] - \frac{1}{N}\sum_{i=1}^{N} g_\theta(x'_i)\Big)^2\Big] + \\ &2\,\mathbb{E}_{P(x'_{1:n}|x)}\Big[\big(f_\theta(x) - \mathbb{E}_{x''}[g_\theta(x'')] - y(x)\big) \\ &\qquad \Big(\mathbb{E}_{x''}[g_\theta(x'')] - \frac{1}{N}\sum_{i=1}^{N} g_\theta(x'_i)\Big)\Big]. \end{aligned} \tag{24}$$

Since $\mathbb{E}_{x''}[g_\theta(x'')] = \mathbb{E}_{x'_1,\cdots,x'_N}[\frac{1}{N}\sum_{i=1}^{N} g_\theta(x'_i)]$, the last term in Equation (24) is zero, and we have

$$\hat{\mathcal{L}}_\theta(x) = \mathcal{L}_\theta(x) + \mathbb{V}_{P(x'_{1:n}|x)}[\frac{1}{N}\sum_{i=1}^{N} g_\theta(x'_i)], \tag{25}$$

where $\mathbb{V}$ denotes the variance operator. If the $x'_1, \cdots, x'_N$ values are sampled in an i.i.d. manner, Equation (25) simplifies further to

$$\hat{\mathcal{L}}_\theta(x) = \mathcal{L}_\theta(x) + \frac{1}{N}\mathbb{V}_{P(x'|x)}[g_\theta(x')]. \tag{26}$$

The induced excess variance in Equation (26) can bias the optimal solution. As a result, optimizing the approximated loss will prefer smoother solutions over all $x'_1, \cdots, x'_N$ samples. It is worth noting that this bias is mostly harmful due to its *parametrized nature*; the only link through which this bias can offset the optimal solution is its *dependency on* θ. This is in contrast to any non-parametrized stochasticity in the y term of Equation (17). Non-parameterized terms cannot offset the optimal solutions, since stochastic gradient descent methods are indifferent to them.

3. Potential Solutions

Based on Equation (26), the induced bias in the solution has a direct relationship with the stochasticity of the $P(x'|x)$ distribution. If we were to sample the (x, x') pairs deterministically, the excess variance in Equation (26) would disappear. However, this results in modifying the problem conditions. Next, we introduce three potential solutions to this problem: the *deterministic sampling approaches*, the *double-sampling trick*, and the *delayed target method* which is based upon the method of learning from temporal differences (Sutton, 1984).

3.1. The Deterministic Sampling Approaches

One approach to eliminate the excess variance term in Equation (25), is to sample the $(x'_1, \cdots, x'_N)$ tuple in a way that $P(x'_{1:n}|x)$ would be a point mass distribution at a fixed T^x tuple. This way, $P(x'_{1:n}|x)$ yields a zero excess variance:

$$\mathbb{V}_{P(x'_{1:n}|x)}[\frac{1}{N}\sum_{i=1}^{N} g_\theta(\mathrm{T}^x_i)] = 0. \tag{27}$$

This induces the following deterministic loss.

$$\hat{\mathcal{L}}^{\text{DET}}_\theta(x) := \left(f_\theta(x) - \frac{1}{N}\sum_{i=1}^{N} g_\theta(\mathrm{T}^x_i) - y(x)\right)^2. \tag{28}$$

Although this approach removes the excess variance term in Equation (25) thanks to its deterministic nature, it biases the optimization loss by re-defining it: $\mathcal{L}_\theta(x) \neq \hat{\mathcal{L}}^{\text{DET}}_\theta(x)$. The choice of the T^x samples can influence the extent of this discrepancy. One reasonable choice is to evenly space the N samples to cover the entire sampling domain as uniformly

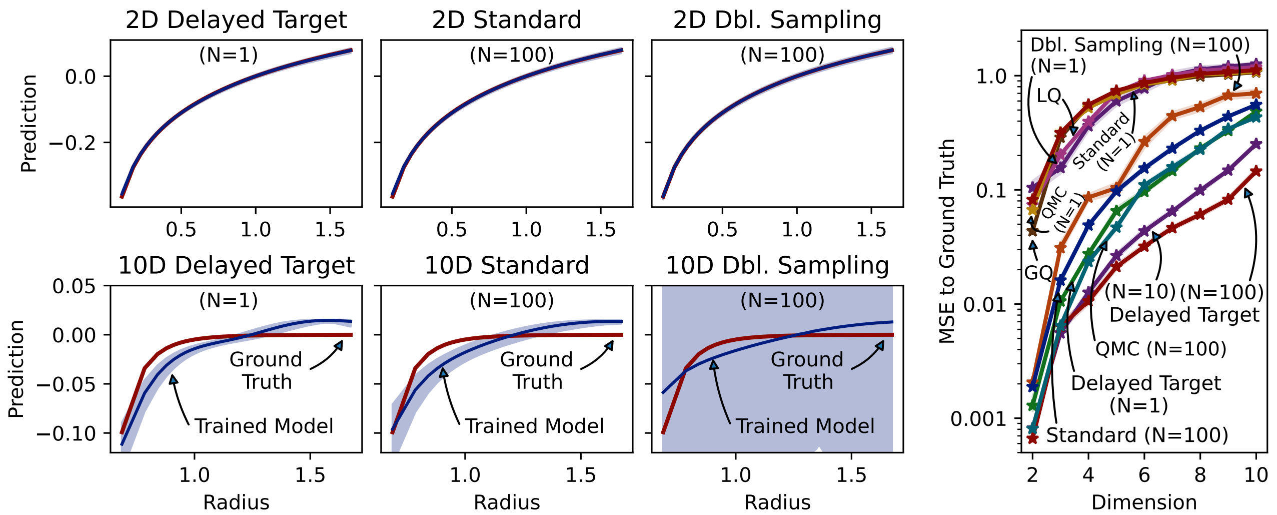

Figure 4. The solution and performance curves in higher-dimensional Poisson problems. *The left panel* shows the solution curves for the delayed target ($N = 1$), the standard ($N = 100$), and the double-sampling ($N = 100$) methods. The top and the bottom rows show 2- and 10-dimensional problems, respectively. In these problems, a single charge is located at the origin, so that the analytical solution is a function of the evaluation point radii $\|x\|$. The horizontal axis shows the evaluation point radii and covers 98% of points within the training volumes. *The right chart* is a performance curve against the problem dimension (lower is better). The normalized MSE values were shown to be comparable. These results suggest that (1) higher dimensions make the problem challenging, and (2) delayed targeting with $N = 1$ is comparable to standard trainings with $N = 100$. GQ and LQ refer to Gaussian and Leja quadrature, respectively, under a Smolyak sparse grid. Sections C.6, C.9, and D.8 of the supplementary material describe the effect of sampling dimension on numerical quadrature and QMC, the effective way of scaling up N for delayed targeting, and the performance evaluation profile, respectively.

as possible. For a demonstration, Figure 3 shows a number of example sets used for applying the divergence theorem to the Poisson problem. Of course, this sampling strategy can be impractical in high-dimensional spaces as the number of samples needed to cover the entire sampling domain grows exponentially with the sampling space dimension. This could be partially ameliorated by the use of QMC methods (Morokoff & Caflisch, 1995).

Numerical quadrature offers another deterministic approach for accurate integral estimation. By choosing specific integration points and weights, they can provably yield accurate integral estimates under certain function classes; for instance, Gaussian quadrature (Gauss, 1814) with N samples can produce exact $(2N-1)$-degree polynomial integrals. However, these methods still suffer from the curse of dimensionality and are more restrictive than the QMC alternatives in their choice of N. This exponential sample requirement can be partially ameliorated by the use of sparse grid methods such as Smolyak's quadrature (Smolyak, 1963).

3.2. The Double-Sampling Trick

If we have two independent x' samples, namely x'_1 and x'_2, we can replace the objective in Equation (22) with

$$\hat{\mathcal{L}}_\theta^{\text{DBL}}(x) = \mathbb{E}_{x'_1,x'_2\sim P(\cdot|x)}\Big[\big(f_\theta(x) - g_\theta(x'_1) - y(x)\big) \big(f_\theta(x) - g_\theta(x'_2) - y(x)\big)\Big]. \tag{29}$$

It is straightforward to show that $\hat{\mathcal{L}}_\theta^{\text{DBL}}(x) = \mathcal{L}_\theta(x)$; the uncorrelation between $g_\theta(x'_1)$ and $g_\theta(x'_2)$ will remove the induced bias on average. However, this approach requires access to two i.i.d. samples, which may not be plausible in many sampling schemes. In particular, Monte-Carlo samplings used in reinforcement learning do not usually afford the learning method with the freedom to choose multiple next samples or the ability to reset to a previous state. Besides reinforcement learning, offline learning using a given collection of samples may make this approach impractical. It is possible to simulate $N = 1$ (and similarly for larger N) in problems of the form $\mathbb{E}_{P(x'|x)}[g_\theta(x')] = y(x)$, such as Examples 2.1 and 2.2, by redefining $\hat{\mathcal{L}}^{\text{DBL}}$ as

$$\hat{\mathcal{L}}_\theta^{\text{DBL}}(x) = \mathbb{E}_{x'_1,x'_2\sim P(\cdot|x)}\Big[\big(g_\theta(x'_1) - y(x)\big) \big(g_\theta(x'_2) - y(x)\big)\Big]. \tag{30}$$

3.3. The Delayed Target Method

This approach replaces the objective in Equation (22) with

$$\mathcal{L}_\theta^{\text{DT}}(x) = \mathbb{E}_{P(x'|x)}\Big[\big(f_\theta(x) - g_{\theta^*}(x') - y(x)\big)^2\Big], \tag{31}$$

where we have $\theta^* := \arg\min_{\tilde{\theta}} \mathcal{L}_{\tilde{\theta}}(x)$. Assuming a complete function approximation set Θ (where $\theta \in \Theta$), we know that θ^* satisfies Equation (1) at all x. Therefore, we have

$$\nabla_\theta \mathcal{L}_\theta(x)\big|_{\theta=\theta^*} = \nabla_\theta \mathcal{L}_\theta^{\text{DT}}(x)\big|_{\theta=\theta^*} = 0. \tag{32}$$

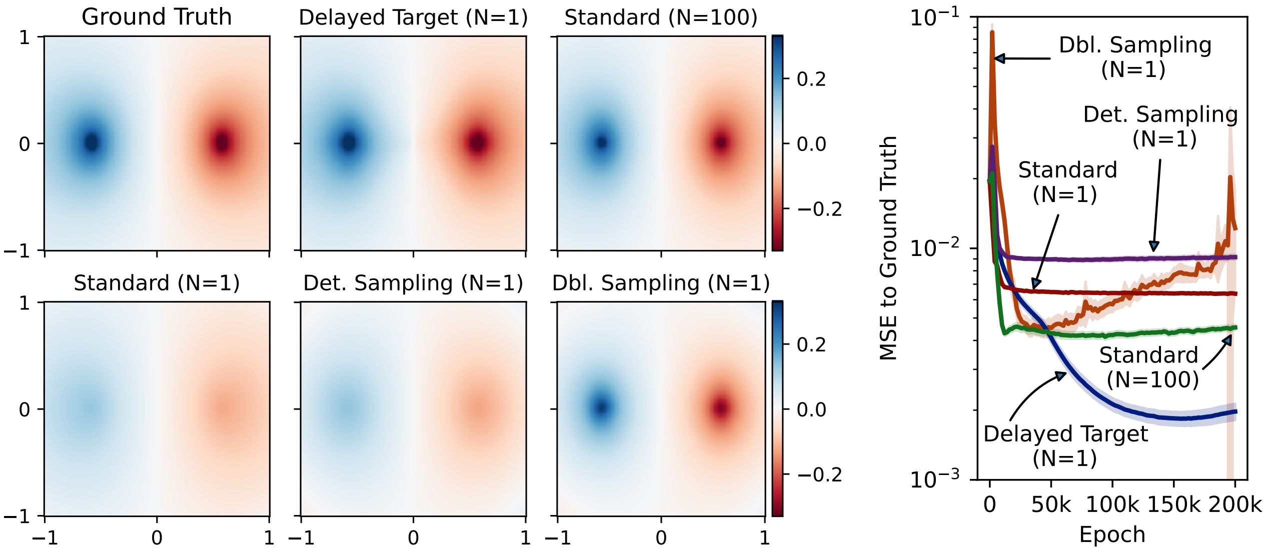

Figure 5. The solution heatmaps and the training curves for different methods to the Maxwell problem. *In the left panel*, we show a single component of the magnetic potentials (A_z) in a 2D slice of the training space with $z = 0$ for visual comparison. *In the right plot*, we show the training curves. The results suggest that (1) the standard and deterministic trainings with $N = 1$ produce overly smooth solutions, and (2) delayed targeting with $N = 1$ is comparable to standard trainings with $N = 100$. Section C.7 of the supplementary material studies the target smoothing and regularization weights of the delayed target method in this problem.

As a result, we can claim

$$\theta^* = \arg\min_\theta \mathbb{E}_x[\mathcal{L}_\theta^{\text{DT}}(x)] = \arg\min_\theta \mathbb{E}_x[\mathcal{L}_\theta(x)]. \quad (33)$$

In other words, optimizing Equation (31) should yield the same solution as optimizing the true objective $\mathcal{L}_\theta(x)$ in Equation (17). Of course, finding θ^* is as difficult as solving the original problem. The simplest heuristic replaces θ^* with a supposedly independent, yet identically valued, version of the latest θ named θ^{Target}, hence the *delayed*, *detached*, and *bootstrapped target* naming conventions:

$$\hat{\mathcal{L}}_\theta^{\text{DT}}(x) = \mathbb{E}_{P(x'_{1:n}|x)}\left[\left(f_\theta(x) - \frac{1}{N}\sum_{i=1}^{N} g_{\theta^{\text{Target}}}(x'_i) - y(x)\right)^2\right]. \quad (34)$$

Our hope would be for this approximation to improve as well as θ over training. The only practical difference between implementing this approach and minimizing the loss in Equation (17) is to use an incomplete gradient for updating θ by detaching the $g(x')$ node from the computational graph in the automatic differentiation software. This naive implementation of the delayed target method can lead to divergence in optimization, as we will show in Section 5 with numerical examples (i.e., Figure 3). Here, we introduce two mitigation factors contributing to the stabilization of such a technique.

Moving Target Stabilization One disadvantage of the aforementioned technique is that it does not define a global optimization objective; even the average target for $f_\theta(x)$ (i.e., $\mathbb{E}_{P(x'|x)}\left[g_{\theta^{\text{Target}}}(x')\right] + y(x)$) changes throughout the training. Therefore, a naive implementation can risk training instability or even divergence thanks to the moving targets.

To alleviate the fast-moving targets issue, prior work suggested fixing the target network for many time-steps (Mnih et al., 2015). This causes the training trajectory to be divided into a number of episodes, where the target is locally constant and the training is therefore locally stable in each episode. Alternatively, this stabilization can be implemented continuously using Polyak averaging (Fujimoto et al., 2018); instead of fixing the target network for a window of T steps, the target parameters θ^{Target} can be updated slowly with the following rule:

$$\theta^{\text{Target}} \leftarrow \tau\theta^{\text{Target}} + (1-\tau)\theta. \quad (35)$$

This exponential moving average defines a corresponding stability window of $T = O(1/(1-\tau))$.

Prior Imposition for Highly Stochastic Targets In certain instances, the $\frac{1}{N}\sum_{i=1}^{N} g_{\theta^{\text{Target}}}(x'_i) + y(x)$ target in Equation (31) can be excessively stochastic, leading to divergence in the training of the delayed target model. For instance, based on Equations (6), (7), (8), (9), and (10) for the Poisson problem, we can write $g_\theta(x_i) = (M-1)f_\theta(x_i)$. Therefore, we can analyze the target variance as

$$\mathbb{V}[\frac{1}{N}\sum_{i=1}^{N} g_{\theta^{\text{Target}}}(x'_i) + y(x) \mid x] = \frac{(M-1)^2}{N}\mathbb{V}[f_{\theta^{\text{Target}}}(x') \mid x] + \mathbb{V}[y(x) \mid x]. \quad (36)$$

Ideally, $M \to \infty$ in order for Equation (5) to hold. Setting arbitrarily large M will lead to unbounded target variances in Equation (36), which can slow down the convergence of the training or result in divergence. In particular, such unbounded variances can cause the main and the target models to drift away from each other, leading to incorrect solutions as we will show in Figure 3 for example.

To prevent this drift, one technique is to impose a Bayesian prior on the main and the target models. Therefore, to discourage this divergence phenomenon, we regularize the delayed target objective in Equation (37) and replace it with

$$\hat{\mathcal{L}}_{\theta}^{\text{DTR}} := \hat{\mathcal{L}}_{\theta}^{\text{DT}}(x) + \lambda \cdot (f_{\theta}(x) - f_{\theta^{\text{Target}}}(x))^2. \tag{37}$$

A formal description of the regularized delayed targeting process is given in Algorithm 1, which covers both the moving target stabilization and the Bayesian prior imposition.

4. Theoretical Results

The double-sampling method and all the deterministic sampling variants use complete gradients for optimization. Thus, they enjoy all the classical convergence guarantees and computational complexity analyses pertaining to the traditional stochastic gradient descent. Essentially, all these methods aim to solve for the fixed point of Equation 1.

However, the delayed target method is different. In fact, the delayed target method can be presented as an instance of stochastic approximation to solve a slightly different fixed point problem:

Theorem 4.1. *Following the assumptions and notation defined in Section B.1 of the supplementary material, notably*

(1) a linear function approximation $f_\theta(x) = \phi(x)^{\mathrm{T}}\theta$,

(2) appropriate η_t learning rates such that $\sum_{t=0}^{\infty} \eta_t = \infty$ and $\sum_{t=0}^{\infty} \eta_t^2 \leq \infty$,

(3) a small $\tau \to 0$ with $\lambda = 0$ and $N = 1$,

(4) $\mathcal{U}$ denoting the training update operator, and

(5) Π being a projection operator to the function approximation class,

the delayed target method is an instance of stochastic approximation (Robbins & Monro, 1951; Kiefer & Wolfowitz, 1952) and converges to the fixed point of the $\Pi\mathcal{U}$ composite operator in the following equation:

$$\Phi\theta_{DT}^{*} = \Pi\mathcal{U}\ \Phi\theta_{DT}^{*}. \tag{38}$$

This is in contrast to the standard training method, which solves for the fixed point of the $\mathcal{U}$ update operator under the same conditions:

$$\Phi\theta^{*} = \mathcal{U}\ \Phi\theta^{*}. \tag{39}$$

Also, assuming that f^* is the fixed point to the $\mathcal{U}$ operator, the approximation error for the delayed target method under these conditions can be upper-bounded as

$$\mathbb{E}_{x\sim P}[(f_{\theta_{DT}^{*}}(x) - f^{*}(x))^2] \leq \\ \frac{1}{1-\sigma_{\mathbf{P}^{|}\Lambda}} \mathbb{E}_{x\sim P}\big[(\Pi\ f^{*}(x) - f^{*}(x))^2\big]. \tag{40}$$

For the detailed statement and proof of Theorem 4.1 as well as the rest of the assumptions and notation (e.g., Φ, σ, $\mathbf{P}^{|}$, Λ, and f^*), see Section B of the supplementary material. Since the delayed target method is an instance of stochastic approximation, its total computational cost to reach an optimization error of ϵ in a d-dimensional parameter space can be $O(d/\epsilon)$, whereas the standard training method may cost $O(Nd\log(1/\epsilon))$ to achieve the same goal. This is discussed further in Section B.3 of the supplementary material.

Of course, with a non-linear function approximation class, the delayed target method may not converge to reasonable solutions; Figure 3 demonstrates such an incorrect solution example. Our setup is more general than reinforcement learning, where such effects have been a topic of research for many decades (Baird, 1995; Boyan & Moore, 1995; Gordon, 1995; Tsitsiklis & Van Roy, 1997; 1996; Bertsekas, 1995; Dayan, 1992; Bertsekas & Tsitsiklis, 1996).

5. Experiments

We examine solving three problems. First, we solve a Poisson problem with singular charges using the divergence theorem as a proxy for learning. In Section 5.1, we define a 2D Poisson problem with three unit Dirac-delta charges at $[0, 0]$, $[-0.5, -0.5]$, and $[0.5, 0.5]$. Figures 1, 2, and 3 demonstrate the potential solutions to this problem. We also study higher-dimensional Poisson problems with a unit charge at the origin in Figure 4. Our second example in Section 5.2 looks at finding the magnetic potentials and fields around a current circuit. The current circuit consists of four wire segments and defines a singular J current density profile. Finally, in Section 5.3 we consider a Smoluchowski coagulation problem to simulate particle evolution dynamics. We designed the coagulation kernel K to induce non-trivial solutions in our solution intervals.

We employed multi-layer perceptrons as our deep neural networks, using 64 hidden neural units in each layer, and either the SiLU or $\tanh$ activation functions. We trained our networks using the Adam (Kingma & Ba, 2014) variant of the stochastic gradient descent algorithm under a learning rate of 0.001. We afforded each method 1000 point evaluations for each epoch. A wealth of ablation studies with more datasets and other experimental details were left to Section C of the supplementary material.

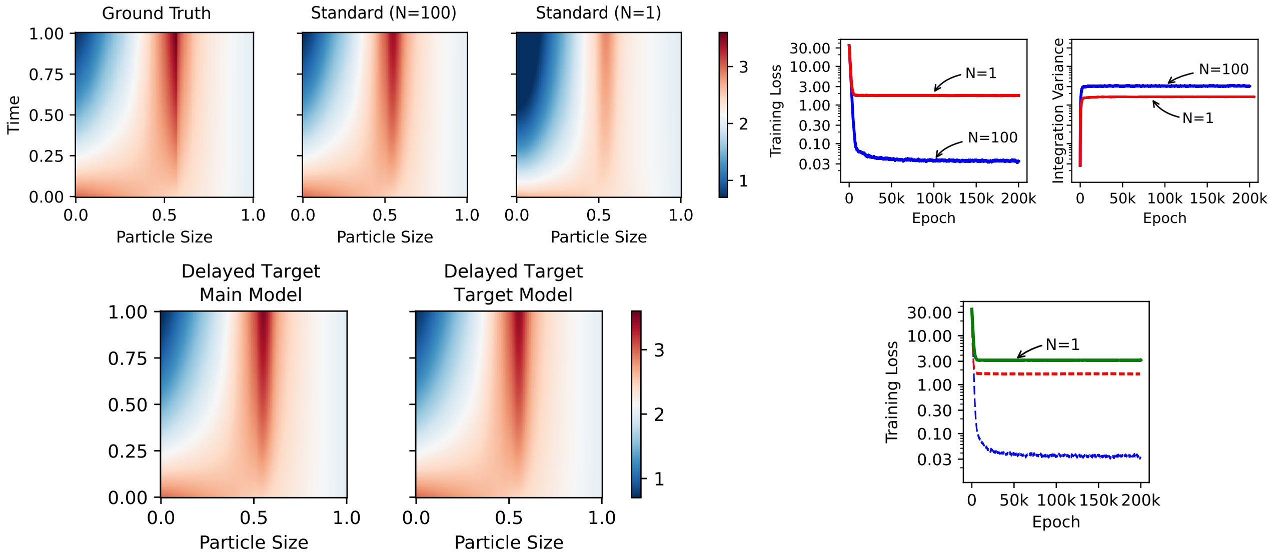

Figure 6. Training results on the Smoluchowski coagulation problem. The top left panel shows the ground truth solution, along with the standard $N = 100$ and $N = 1$ solutions minimizing the $\hat{\mathcal{L}}_\theta(x)$ in Equation (26). The training loss and the integration variance represent the $\hat{\mathcal{L}}_\theta(x)$ and $\mathbb{V}_{P(x'|x)}[g_\theta(x')]$ quantities in Equation (26). The top right figure shows the training curve for both of the standard trainings. The bottom left panel shows the delayed target solution heatmaps using $N = 1$ sample with its training curve next to it.

5.1. The Poisson Problem with Singular Charges

To show the solution bias, we first train two models: one with $N = 100$ samples per sphere, and another one with only $N = 1$ sample per sphere. These models represent a baseline for later comparisons. Based on Equation (26), the induced solution bias should be lower in the former scenario. Figure 1 shows the solution defined by these models along with the analytical solution and their respective training curves. The model trained with high estimation variance derives an overly smooth solution. We hypothesize that this is due to the excess variance in the loss. This hypothesis is confirmed by matching the training loss and the excess variance curves; the training loss of the model with $N = 1$ is lower bounded by its excess variance, although it successfully finds a solution with a smaller excess variance than the $N = 100$ model. An alternative capable of producing similar quality solutions with $N = 1$ sample would be ideal.

To investigate the effect of highly stochastic targets on delayed target models, Figure 3 shows the training results with both $M = 100$ and $M = 10$. The former is unstable, while the latter is stable; this confirms the influence of M in the convergence of the delayed target trainings. Furthermore, when this divergence happens, a clear drift between the main and the target models can be observed. Figure (3) shows that imposing the Bayesian prior of Equation (37) can lead to training convergence even with a larger $M = 1000$, which demonstrates the utility of our proposed solution.

We also investigated the performance of the deterministic and double-sampling techniques in this problem. Figure 2 shows these results when $N = 1$ and $N = 100$ samples are used for integral estimation. With $N = 1$, the training with the deterministic sampling approach is stable and yields similar results to those seen in Figure (1). The double-sampling trick, on the other hand, exhibits unstable trainings and suboptimal solutions. We suspect that (a) the singular nature of the analytical solution, and (b) the stochasticity profile of the training loss function $\hat{\mathcal{L}}_\theta^{\mathrm{DBL}}(x)$ in Equation (29) are two of the major factors contributing to this outcome. With $N = 100$, both the deterministic and double-sampling trainings yield stable training curves and better solutions. This suggests that both methods can still be considered viable options for training integro-differential PINNs, conditioned on that the specified N is large enough for these methods to train stably and well.

The regularized delayed target training with $N = 1$ sample is also shown in the training curves of Figure 2 for easier comparison. The delayed target method yields better performance than the deterministic or double-sampling in this problem. This may seemingly contradict the fact that the double-sampling method enjoys better theoretical guarantees than the delayed target method since it optimizes a complete gradient. However, our results are consistent with recent findings in off-policy reinforcement learning; even in deterministic environments where the application of the double-sampling method can be facilitated with a single sample, incomplete gradient methods (e.g., TD-learning) may still be preferable over the full gradient methods (e.g., double-sampling) (Saleh & Jiang, 2019; Fujimoto et al., 2022; Yin et al., 2022; Chen et al., 2021). Intuitively, incomplete gradient methods detach parts of the gradient, depriving the optimizer from exercising full control over

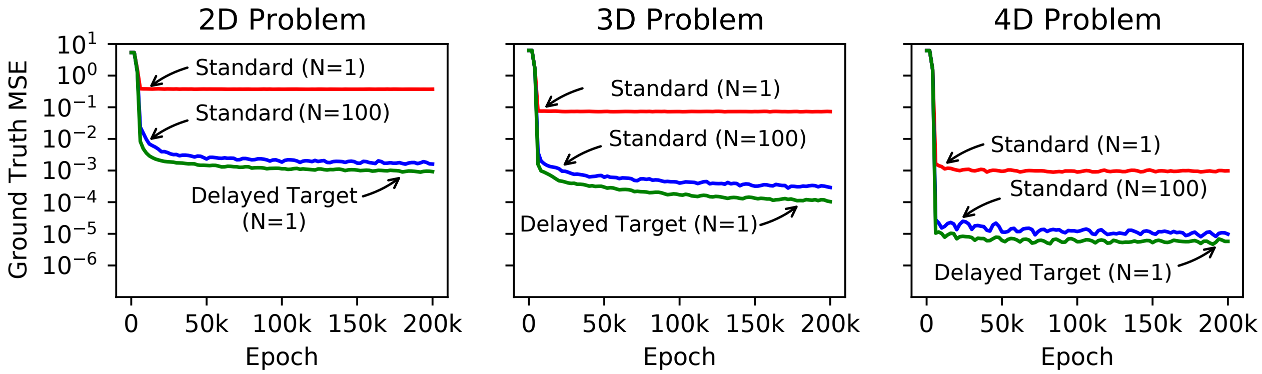

Figure 7. The solution mean squared error to the ground truth in the 2, 3, and 4-dimensional Smoluchowski coagulation problem. The vertical axis shows the solution error, and the horizontal axis shows the training epochs. The standard solutions were trained by the ordinary MSE loss $\mathcal{L}_\theta(x)$ in Equation (17) with $N = 1$ and $N = 100$ samples. The delayed target solution used $N = 1$ sample, yet produced slightly better results than the standard method with $N = 100$.

the decent direction and make it avoid over-fitting. In other words, incomplete gradient methods can be viewed as a middle ground between zero-order and first-order optimization and may be preferable over both of them.

Figure 4 also studies the effect of problem dimensionality on our methods. The results confirm that the problem becomes significantly more difficult with higher dimensions. However, the delayed target solutions maintain comparable quality to standard trainings with large N. Gaussian and Leja numerical quadrature seem to be less effective in this problem. QMC methods, on the other hand, certainly improve upon the standard i.i.d. estimators. The delayed target with $N = 1$ performs similarly to the standard and QMC methods with $N = 100$, and can be improved further by increasing N (see Section C.9 of the supplementary material on scaling up N effectively in the delayed target method).

5.2. The Maxwell Problem with a Wired Circuit

Figure 5 shows the training results for the Maxwell problem. The results suggest that the standard and the deterministic trainings with small N produce overly smooth solutions. The double-sampling method with small N improves the solution quality at first but has difficulty maintaining a stable improvement. However, delayed targeting with small N seems to produce comparable solutions to the standard training with large N.

5.3. The Smoluchowski Coagulation Problem

Figure 6 shows the training results for the Smoluchowski coagulation problem. Similar to the results in Figure 1, the standard training using $N = 1$ sample for computing the residual summations leads to a biased and sub-optimal solution. However, the standard training with $N = 100$ samples suffers less from the effect of bias. The delayed target solution using only $N = 1$ sample produces comparable solution quality to the standard evaluation with $N = 100$ and is not bottlenecked by the integration variance. Figure 7 compares the solution quality for each of the standard and delayed target methods under different problem dimensions. The results suggest that the delayed target solution maintains its quality even in higher dimensional problems, where the excess variance issue leading to biased solutions may be more pronounced.

6. Discussion

In this work, we investigated the problem of learning PINNs in partial integro-differential equations. We presented a general framework for the problem of learning from integral losses and theoretically showed that naive approximations of the parametrized integrals lead to biased loss functions due to the induced excess variance term in the optimization objective. We confirmed the existence of this issue in numerical simulations. Then, we studied three potential solutions to account for this issue, and we found the delayed target method to perform best in a wide class of problems. Our numerical results support the utility of this method on three classes of problems, (1) Poisson problems with singular charges and up to 10 dimensions, (2) an electromagnetic problem under a Maxwell equation, and (3) a Smoluchowski coagulation problem. The limitations of our work include its narrow scope in learning PINNs; this work could have broader applications in other areas of machine learning. Also, future work should consider the applications of the delayed target method to more problem classes in both scientific and traditional machine learning. Developing adaptive processes for setting each method's hyper-parameters, such as the training batch-sizes and regularization weights in the delayed target method, and combining importance sampling, numerical quadrature, or QMC techniques with our methods are two other worthwhile future endeavors.

Acknowledgements

This work used GPU resources at the Delta supercomputer of the National Center for Supercomputing Applications through Allocation CIS220111 from the Advanced Cyberinfrastructure Coordination Ecosystem: Services and Support (ACCESS) program (Boerner et al., 2023), which is supported by National Science Foundation grants #2138259, #2138286, #2138307, #2137603, and #2138296.

Impact Statement

This work provides foundational theoretical results and builds upon methods for training neural PDE solvers within the area of scientific learning. Scientific learning methods and neural PDE solvers can provide valuable models for a solving range of challenging applications in additive manufacturing (Zhu et al., 2021; Niaki et al., 2021; Henkes et al., 2022), robotics (Sun et al., 2022), high-speed flows (Mao et al., 2020), weather-forecasting (Mammedov et al., 2021), finance systems (Bai et al., 2022) chemistry (Ji et al., 2021), computational biology (Lagergren et al., 2020), and heat transfer and thermodynamics (Cai et al., 2021).

Although many implications could result from the application of scientific learning, in this work we focused especially on settings where precision, singular inputs, and compatibility with partial observations are required for solving the PDEs. Our work particularly investigated methods for learning PDEs with integral forms and provided effective solutions for solving them. Such improvements could help democratize the usage of physics-informed networks in applications where independent observations are difficult or expensive to obtain, and the inter-sample relationships and constraints may contain the majority of the training information. Such problems may be challenging and the trained models are usually less precise than the traditional solvers. These errors can propagate to any downstream analysis and decision-making processes and result in significant issues. Other negative consequences of this work could include weak interpretability of the trained models, increased costs for re-training the models given varying inputs, difficulty in estimating the performance of such trained models, and the existence of unforeseen artifacts in the trained models (Wang et al., 2021).

To mitigate the risks, we encourage further research to develop methods to provide guarantees and definitive answers about model behaviors. In other words, a general framework for making guaranteed statements about the behavior of the trained models is missing. Furthermore, more efficient methods for training such models on a large variety of inputs should be prioritized for research. Also, a better understanding of the pathology of neural solvers is of paramount concern to use these models safely and effectively.

References

Arazo, E., Ortego, D., Albert, P., O'Connor, N. E., and McGuinness, K. Pseudo-labeling and confirmation bias in deep semi-supervised learning. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8. IEEE, 2020.

Bai, Y., Chaolu, T., and Bilige, S. The application of improved physics-informed neural network (ipinn) method in finance. *Nonlinear Dynamics*, 107(4):3655–3667, 2022.

Baird, L. Residual algorithms: Reinforcement learning with function approximation. In *Machine Learning Proceedings 1995*, pp. 30–37. Elsevier, 1995.

Bern, Z., Dixon, L., and Kosower, D. A. Dimensionally-regulated pentagon integrals. *Nuclear Physics B*, 412(3): 751–816, 1994. ISSN 0550-3213.

Bertsekas, D. P. A counterexample to temporal differences learning. *Neural computation*, 7(2):270–279, 1995.

Bertsekas, D. P. and Tsitsiklis, J. N. *Neuro-Dynamic Programming*. Athena Scientific, Belmont, MA, 1996.

Boerner, T. J., Deems, S., Furlani, T. R., Knuth, S. L., and Towns, J. Access: Advancing innovation: Nsf's advanced cyberinfrastructure coordination ecosystem: Services & support. In *Practice and Experience in Advanced Research Computing*, pp. 173–176. 2023.

Boyan, J. and Moore, A. W. Generalization in reinforcement learning: Safely approximating the value function. *Advances in neural information processing systems*, pp. 369–376, 1995.

Cabré, X. and Roquejoffre, J.-M. The influence of fractional diffusion in fisher-kpp equations. *Communications in Mathematical Physics*, 320(3):679–722, 2013.

Caffarelli, L. and Vasseur, A. The de giorgi method for regularity of solutions of elliptic equations and its applications to fluid dynamics. *Discrete Contin. Dyn. Syst. Ser. S*, 3(3):409–427, 2010a.

Caffarelli, L., Chan, C. H., and Vasseur, A. Regularity theory for parabolic nonlinear integral operators. *Journal of the American Mathematical Society*, 24(3):849–869, 2011.

Caffarelli, L. A. and Vasseur, A. Drift diffusion equations with fractional diffusion and the quasi-geostrophic equation. *Annals of Mathematics*, pp. 1903–1930, 2010b.

Caflisch, R. E. Monte carlo and quasi-monte carlo methods. *Acta numerica*, 7:1–49, 1998.

Cai, S., Wang, Z., Wang, S., Perdikaris, P., and Karniadakis, G. E. Physics-informed neural networks for heat transfer problems. *Journal of Heat Transfer*, 143(6), 2021.

Carrillo, J. A., DiFrancesco, M., Figalli, A., Laurent, T., and Slepčev, D. Global-in-time weak measure solutions and finite-time aggregation for nonlocal interaction equations. 2011.

Chapman, S. J., Rubinstein, J., and Schatzman, M. A mean-field model of superconducting vortices. *European Journal of Applied Mathematics*, 7(2):97–111, 1996.

Chen, Y., Xu, L., Gulcehre, C., Paine, T. L., Gretton, A., de Freitas, N., and Doucet, A. On instrumental variable regression for deep offline policy evaluation. *arXiv preprint arXiv:2105.10148*, 2021.

Constantin, P. Euler equations, navier-stokes equations and turbulence. In *Mathematical Foundation of Turbulent Viscous Flows: Lectures given at the CIME Summer School held in Martina Franca, Italy, SEptember 1-5, 2003*, pp. 1–43. Springer, 2005.

Dayan, P. The convergence of td (λ) for general λ. *Machine learning*, 8(3-4):341–362, 1992.

Elgart, A. and Schlein, B. Mean field dynamics of boson stars. *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, 60(4):500–545, 2007.

Evans, M. and Swartz, T. Methods for approximating integrals in statistics with special emphasis on bayesian integration problems. *Statistical science*, pp. 254–272, 1995.

Fujimoto, S., Van Hoof, H., and Meger, D. Addressing function approximation error in actor-critic methods. *arXiv preprint arXiv:1802.09477*, 2018.

Fujimoto, S., Meger, D., Precup, D., Nachum, O., and Gu, S. S. Why should i trust you, bellman? the bellman error is a poor replacement for value error. *arXiv preprint arXiv:2201.12417*, 2022.

Gauss, C. F. *Methodus nova integralium valores per approximationem inveniendi*. 1814.

Ghaffari, S., Saleh, E., Forsyth, D., and Wang, Y.-X. On the importance of firth bias reduction in few-shot classification. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=DNRADop4ksB.

Giacomin, G. and Lebowitz, J. L. Phase segregation dynamics in particle systems with long range interactions. i. macroscopic limits. *Journal of statistical Physics*, 87: 37–61, 1997.

Givan, R., Dean, T., and Greig, M. Equivalence notions and model minimization in Markov decision processes. *Artificial Intelligence*, 147(1):163–223, 2003.

Gordon, G. J. Stable function approximation in dynamic programming. In *Proceedings of the twelfth international conference on machine learning*, pp. 261–268, 1995.

Guo, L., Wu, H., Yu, X., and Zhou, T. Monte carlo fpinns: Deep learning method for forward and inverse problems involving high dimensional fractional partial differential equations. *Computer Methods in Applied Mechanics and Engineering*, 400:115523, 2022. ISSN 0045-7825.

Henkes, A., Wessels, H., and Mahnken, R. Physics informed neural networks for continuum micromechanics. *Computer Methods in Applied Mechanics and Engineering*, 393:114790, 2022.

Hinton, G., Vinyals, O., Dean, J., et al. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7), 2015.

Humphries, N. E., Queiroz, N., Dyer, J. R., Pade, N. G., Musyl, M. K., Schaefer, K. M., Fuller, D. W., Brunnschweiler, J. M., Doyle, T. K., Houghton, J. D., et al. Environmental context explains lévy and brownian movement patterns of marine predators. *Nature*, 465 (7301):1066–1069, 2010.

Jagtap, A. D., Kharazmi, E., and Karniadakis, G. E. Conservative physics-informed neural networks on discrete domains for conservation laws: Applications to forward and inverse problems. *Computer Methods in Applied Mechanics and Engineering*, 365:113028, 2020.

Ji, W., Qiu, W., Shi, Z., Pan, S., and Deng, S. Stiff-pinn: Physics-informed neural network for stiff chemical kinetics. *The Journal of Physical Chemistry A*, 125(36): 8098–8106, 2021.

Jiang, N., Kulesza, A., and Singh, S. Abstraction Selection in Model-based Reinforcement Learning. In *Proceedings of the 32nd International Conference on Machine Learning*, pp. 179–188, 2015.

Jong, N. K. and Stone, P. State abstraction discovery from irrelevant state variables. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, pp. 752–757, 2005.

Kharazmi, E., Zhang, Z., and Karniadakis, G. E. Variational physics-informed neural networks for solving partial differential equations. *arXiv preprint arXiv:1912.00873*, 2019.

Kharazmi, E., Zhang, Z., and Karniadakis, G. E. hp-vpinns: Variational physics-informed neural networks with domain decomposition. *Computer Methods in Applied Mechanics and Engineering*, 374:113547, 2021.

Kiefer, J. and Wolfowitz, J. Stochastic estimation of the maximum of a regression function. *The Annals of Mathematical Statistics*, pp. 462–466, 1952.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Lagergren, J. H., Nardini, J. T., Baker, R. E., Simpson, M. J., and Flores, K. B. Biologically-informed neural networks guide mechanistic modeling from sparse experimental data. *PLoS computational biology*, 16(12):e1008462, 2020.

Lakshmikantham, V. *Theory of integro-differential equations*, volume 1. CRC press, 1995.

Laskin, N. Fractional quantum mechanics and lévy path integrals. *Physics Letters A*, 268(4-6):298–305, 2000.

Laskin, N. Fractional schrödinger equation. *Physical Review E*, 66(5):056108, 2002.

Lee, D.-H. et al. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, pp. 896, 2013.

Li, K., Tang, K., Wu, T., and Liao, Q. D3m: A deep domain decomposition method for partial differential equations. *IEEE Access*, 8:5283–5294, 2019.

Li, L. A worst-case comparison between temporal difference and residual gradient with linear function approximation. In *Proceedings of the 25th international conference on machine learning*, pp. 560–567. ACM, 2008.

Li, L. *A unifying framework for computational reinforcement learning theory*. PhD thesis, Rutgers, The State University of New Jersey, 2009.

Li, L., Walsh, T. J., and Littman, M. L. Towards a unified theory of state abstraction for MDPs. In *Proceedings of the 9th International Symposium on Artificial Intelligence and Mathematics*, pp. 531–539, 2006.

Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., and Anandkumar, A. Neural operator: Graph kernel network for partial differential equations. *arXiv preprint arXiv:2003.03485*, 2020.

Lieb, E. H. and Yau, H.-T. The chandrasekhar theory of stellar collapse as the limit of quantum mechanics. *Communications in mathematical physics*, 112(1):147–174, 1987.

Lu, G. The peierls—nabarro model of dislocations: a venerable theory and its current development. In *Handbook of Materials Modeling: Methods*, pp. 793–811. Springer, 2005.

Mammedov, Y. D., Olugu, E. U., and Farah, G. A. Weather forecasting based on data-driven and physics-informed reservoir computing models. *Environmental Science and Pollution Research*, pp. 1–14, 2021.

Mao, Z., Jagtap, A. D., and Karniadakis, G. E. Physics-informed neural networks for high-speed flows. *Computer Methods in Applied Mechanics and Engineering*, 360:112789, 2020.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540): 529–533, 2015.

Modest, M. F. and Mazumder, S. *Radiative heat transfer*. Academic press, 2021.

Morokoff, W. J. and Caflisch, R. E. Quasi-monte carlo integration. *Journal of computational physics*, 122(2): 218–230, 1995.

Niaki, S. A., Haghighat, E., Campbell, T., Poursartip, A., and Vaziri, R. Physics-informed neural network for modelling the thermochemical curing process of composite-tool systems during manufacture. *Computer Methods in Applied Mechanics and Engineering*, 384:113959, 2021.

Nolan, J. P. Fitting data and assessing goodness-of-fit with stable distributions. *Applications of Heavy Tailed Distributions in Economics, Engineering and Statistics, Washington DC*, 1999.

NVIDIA. Physics informed neural networks in modulus, May 2024. URL https://docs.nvidia.com/deeplearning/modulus/modulus-v2209/user_guide/theory/phys_informed.html.

Pham, H., Dai, Z., Xie, Q., and Le, Q. V. Meta pseudo labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11557–11568, 2021.

Raissi, M., Perdikaris, P., and Karniadakis, G. E. 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.

Ravindran, B. *An algebraic approach to abstraction in reinforcement learning*. PhD thesis, University of Massachusetts Amherst, 2004.

Reynolds, A. M. and Rhodes, C. J. The lévy flight paradigm: random search patterns and mechanisms. *Ecology*, 90(4): 877–887, 2009.

Robbins, H. and Monro, S. A stochastic approximation method. *The annals of mathematical statistics*, pp. 400–407, 1951.

Ros Oton, X. Integro-differential equations: Regularity theory and pohozaev identities. 2014.

Saleh, E. and Jiang, N. Deterministic bellman residual minimization. In *Proceedings of Optimization Foundations for Reinforcement Learning Workshop at NeurIPS*, 2019.

Schoknecht, R. and Merke, A. Td (0) converges provably faster than the residual gradient algorithm. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pp. 680–687, 2003.

Shukla, K., Jagtap, A. D., and Karniadakis, G. E. Parallel physics-informed neural networks via domain decomposition. *Journal of Computational Physics*, 447:110683, 2021.

Smolyak, S. A. Quadrature and interpolation formulas for tensor products of certain classes of functions. In *Doklady Akademii Nauk*, volume 148, pp. 1042–1045. Russian Academy of Sciences, 1963.

Sun, W., Akashi, N., Kuniyoshi, Y., and Nakajima, K. Physics-informed recurrent neural networks for soft pneumatic actuators. *IEEE Robotics and Automation Letters*, 7(3):6862–6869, 2022.

Sutton, R. S. *Temporal credit assignment in reinforcement learning*. University of Massachusetts Amherst, 1984.

Sutton, R. S. Learning to predict by the methods of temporal differences. *Machine learning*, 3(1):9–44, 1988.

Tian, Y., Chen, X., and Ganguli, S. Understanding self-supervised learning dynamics without contrastive pairs. In *International Conference on Machine Learning*, pp. 10268–10278. PMLR, 2021.

Toland, J. F. The peierls–nabarro and benjamin–ono equations. *Journal of functional analysis*, 145(1):136–150, 1997.

Tsitsiklis, J. N. and Van Roy, B. Feature-based methods for large scale dynamic programming. *Machine Learning*, 22(1):59–94, 1996.

Tsitsiklis, J. N. and Van Roy, B. An analysis of temporal-difference learning with function approximation. *IEEE Transactions on Automatic Control*, 42(5), 1997.

Viswanathan, G. M., Afanasyev, V., Buldyrev, S. V., Murphy, E. J., Prince, P. A., and Stanley, H. E. Lévy flight search patterns of wandering albatrosses. *Nature*, 381(6581): 413–415, 1996.

Wang, J. L., Curtis, J. H., Riemer, N., and West, M. Learning coagulation processes with combinatorial neural networks. *Journal of Advances in Modeling Earth Systems*, pp. e2022MS003252, 2022a.

Wang, S., Teng, Y., and Perdikaris, P. Understanding and mitigating gradient flow pathologies in physics-informed neural networks. *SIAM Journal on Scientific Computing*, 43(5):A3055–A3081, 2021.

Wang, S., Sankaran, S., and Perdikaris, P. Respecting causality is all you need for training physics-informed neural networks. *arXiv preprint arXiv:2203.07404*, 2022b.

Wei, W., Zhou, B., Połap, D., and Woźniak, M. A regional adaptive variational pde model for computed tomography image reconstruction. *Pattern Recognition*, 92:64–81, 2019.

Weinan, E. Dynamics of vortex liquids in ginzburg-landau theories with applications to superconductivity. *Physical Review B*, 50(2):1126, 1994.

Yin, S., Luo, T., Liu, P., and Xu, Z.-Q. J. An experimental comparison between temporal difference and residual gradient with neural network approximation. *arXiv preprint arXiv:2205.12770*, 2022.

Zhu, Q., Liu, Z., and Yan, J. Machine learning for metal additive manufacturing: predicting temperature and melt pool fluid dynamics using physics-informed neural networks. *Computational Mechanics*, 67:619–635, 2021.

A. Probabilistic and Mathematical Notation

We denote expectations with the $\mathbb{E}_{P(z)}[h(z)] := \int_z h(z)P(z)\mathrm{d}z$, and variances with the $\mathbb{V}_{P(z)}[h(z)] := \mathbb{E}_{P(z)}[h(z)^2] - \mathbb{E}_{P(z)}[h(z)]^2$ notation. Note that only the random variable in the subscript (i.e., z) is eliminated after the expectation. The set of samples $\{x'_1, \cdots x'_n\}$ is denoted with $x'_1, \cdots, x'_N$, and we abuse the notation by replacing $x'_1, \cdots, x'_N$ with $x'_{1:n}$ for brevity. Throughout the manuscript, $f_\theta(x)$ denotes the output of a neural network, parameterized by θ, on the input x. The loss functions used for minimization are denoted with the $\mathcal{L}$ notation (e.g., $\mathcal{L}_\theta(x)$). $\nabla U := [\frac{\partial}{\partial x_1}U, \cdots, \frac{\partial}{\partial x_d}U]$ denotes the gradient of a scalar function U, $\nabla \cdot E := \frac{\partial E_1}{\partial x_1} + \cdots + \frac{\partial E_d}{\partial x_d}$ denotes the divergence of the vector field E, and $\nabla^2 U := \nabla \cdot \nabla U$ denotes the Laplacian of the function U. The d-dimensional Dirac-delta function is denoted with δ^d, volumes are denoted with Ω, and surfaces are denoted with $\partial\Omega$. The Gamma function is denoted with Γ, where $\Gamma(n) := (n-1)!$ for integer n. The uniform probability distribution over an area A is denoted with $\mathrm{Unif}(A)$. These operators and notation are summarized in Tables 1 and 2.

Notation	Description
$f_\theta(x)$	The main neural output parameterized by θ
$g_\theta(x)$	Secondary neural output parameterized by θ
$\mathcal{L}$	Generic loss functions representation
$\mathcal{L}_\theta(x)$	Loss $\mathcal{L}$ parametrized by θ evaluated at x
$\hat{\mathcal{L}}$	Generic approximated loss representation
N	Number of samples used for integral estimation
τ	The delayed target Polyak averaging factor in Algorithm 1 of the main paper
λ	The delayed target regularization weight defined in Equation (37)
δ^d	The d-dimensional Dirac-delta function
Ω	Volume representation
$\partial\Omega$	Surface representation
Γ	The Gamma function, where $\Gamma(n) := (n-1)!$ for integer values
$\mathrm{Unif}(Z)$	The uniform probability distribution over the Z set or interval
U	The potential function in the Poisson problem
E	The gradient field in the Poisson problem
ρ	The input charge density in the Poisson problem
A	The magnetic potentials in the Maxwell-Ampere problem
B	The magnetic field in the Maxwell-Ampere problem
J	The current density field in the Maxwell-Ampere problem
I	The current flowing through a plane in the Maxwell-Ampere problem
K	The Smoluchowski coagulation kernel used in Equation (12)
n	Particle densities in the Smoluchowski equation

Table 1. The mathematical notation used throughout the paper.

Notation	Definition	Description
$\nabla \cdot E$	$\frac{\partial E_1}{\partial x_1} + \cdots + \frac{\partial E_d}{\partial x_d}$	Divergence of the E field
$\nabla^2 U$	$\nabla \cdot \nabla U$	Laplacian of the U potential
$\nabla \times A$	$\left[\frac{\partial A_3}{\partial x_2} - \frac{\partial A_2}{\partial x_3}, \quad \frac{\partial A_1}{\partial x_3} - \frac{\partial A_3}{\partial x_1}, \quad \frac{\partial A_2}{\partial x_1} - \frac{\partial A_1}{\partial x_2}\right]^{\mathrm{T}}$	Curl of the 3D A field

Table 2. The differential operators used throughout the paper.

B. Theoretical Results

Given the empirical performance of the delayed target method, some supporting theoretical results may provide more insight into this method. Here, under a linear function approximation class and certain assumptions described in Section B.1, we present the delayed target method as an instance of stochastic approximation to solve a fixed point problem and upper-bound its approximation error in Section B.2. We also compare the computational complexity of the delayed target and standard training methods in Section B.3.

B.1. Definitions and Assumptions

As stated earlier, we will assume a linear function approximation class:

$$f_\theta(x) := \phi(x)^{\mathrm{T}}\theta, \tag{41}$$
$$g_\theta(x) := \psi(x)^{\mathrm{T}}\theta. \tag{42}$$

For convenience in the theoretical derivations, we will assume that the x domain is discretized into K bins. This is commonly known as an abstraction, where a mapping is applied to compress the original continuous input domain $\mathcal{X}$ into some finite abstract space (Li et al., 2006):

$$x \in \mathcal{X} \cong \{x_1, x_2, \cdots, x_K\}. \tag{43}$$

Under this regime, Certainty Equivalence (CE) models were studied to potentially improve the generalization of functions learned within the abstract space (Givan et al., 2003; Ravindran, 2004; Li, 2009; Jong & Stone, 2005; Jiang et al., 2015). Here, we only use an abstraction to better understand the role of the input domain size and make the theoretical derivations easier to follow. We assume $K \to \infty$, that is, K is an infinitely large integer. To be clear, this has no practical impact on our algorithms, as they run in the original continuous domain. We are mainly abstracting the input domain to express the terms in matrix and vector product forms rather than integrals.

Next, we define the following notation:

- d denotes the parameter dimensions:
$$d := \dim(\theta). \tag{44}$$

- Φ denotes the feature matrix of the f function:
$$\Phi := [\phi(x_1), \phi(x_2), \cdots, \phi(x_K)]^{\mathrm{T}} \in \mathbb{R}^{K\times d}. \tag{45}$$

- Ψ denotes the feature matrix of the g function:
$$\Psi := [\psi(x_1), \psi(x_2), \cdots, \psi(x_K)]^{\mathrm{T}} \in \mathbb{R}^{K\times d}. \tag{46}$$

- $\mathbf{D}_P$ denotes the diagonal matrix consisting of the input sampling probabilities:
$$\mathbf{D}_P := \mathrm{diag}([P(x_1), P(x_2), \cdots, P(x_K)]) \in \mathbb{R}^{K\times K}. \tag{47}$$

- $\mathbf{Y}$ denotes the compiled labels for each input:

$$\mathbf{Y} := [y(x_1), y(x_2), \cdots, y(x_K)] \in \mathbb{R}^K. \tag{48}$$

- $\mathbf{P}^{|}$ denotes the conditional sampling distribution in a matrix form:

$$\mathbf{P}^{|} := [P(x' = x_i | x = x_j)]_{i,j} \in \mathbb{R}^{K \times K}. \tag{49}$$

- Assuming M is a square matrix, σ_M denotes the spectral radius of M:

$$\sigma_M := \max_{\|z\|_2 = 1} |z^{\mathrm{T}} M z|. \tag{50}$$

- We denote f^* and g^* to be the perfect solution to Equation 1. In the vector form over the abstract space, they can be represented as $\mathbf{f}^*$ and $\mathbf{g}^*$, respectively:

$$\mathbf{f}^* := [f^*(x_1), \cdots, f^*(x_K)] \in \mathbb{R}^K, \tag{51}$$

$$\mathbf{g}^* := [g^*(x_1), \cdots, g^*(x_K)] \in \mathbb{R}^K. \tag{52}$$

- Continuous functions can be expressed as vectors in the abstracted space. We denote h for an arbitrary function, and represent it as $\mathbf{H}$ in the vector form over the abstract space:

$$\mathbf{H} := [h(x_1), h(x_2), \cdots, h(x_N)]^{\mathrm{T}} \in \mathbb{R}^K. \tag{53}$$

- The weighted L2-norm for $\mathbf{H}$ under the P distribution can be defined as

$$\|\mathbf{H}\|_P := \mathbb{E}_{x \sim P}[h(x)^2]. \tag{54}$$

We will assume distinctive and bounded features and non-zero probability for sampling all input values:

$$\forall x \in \mathcal{X} : \|\phi(x)\| \leq 1, \quad \|\psi(x)\| \leq 1, \quad P(x) > 0, \tag{55}$$

$$\text{rank}(\Phi) = \text{rank}(\Psi) = d. \tag{56}$$

Since $K > d$, for some $\Lambda \in \mathbb{R}^{K \times K}$ we have

$$\Psi = \Lambda \Phi. \tag{57}$$

Note that Λ can be constructed by a simple linear regression of the rows. Since K is much larger than d, this is an over-parameterized setting and such Λ will always exist.

We will assume that the Λ matrix has a sub-unit spectral radius. In other words, we assume $\sigma_{\mathbf{P}^{|}\Lambda} < 1$ for all $K \geq K_{\min}$, where $K_{\min}$ is a constant. This practically means that *g is not "over-powering" f* and that *f has the most control over the delayed target updates*. For instance, the Bellman equation with a $\gamma < 1$ discount factor

$$V_\theta^\pi(x) = \gamma \mathbb{E}_{P(x'|x)}[V_\theta^\pi(x')] + R(x, \pi) \tag{58}$$

satisfies this condition, since it has $\Lambda = \gamma I$ and $\sigma_{\mathbf{P}^{|}\Lambda} = \gamma \sigma_{\mathbf{P}^{|}} < 1$.

Next, we define the projection and update operators:

$$\Pi\, \mathbf{H} := \underset{z = \Phi\theta}{\arg\min} (\mathbf{H} - \Phi\theta)^{\mathrm{T}} \mathbf{D}_P (\mathbf{H} - \Phi\theta), \tag{59}$$

$$\mathcal{U}\, \mathbf{H} := \mathbf{Y} + \mathbf{P}^{|} \Lambda \mathbf{H}. \tag{60}$$

We can abuse the notation, and express these operators in the original domain as well:

$$\Pi\, h := \underset{\substack{z_{\theta'} \\ \text{s.t. } \forall x:\, z_{\theta'}(x) = \phi(x)^{\mathrm{T}}\theta'}}{\arg\min} \mathbb{E}_{x \sim P}[(h(x) - z_{\theta'}(x))^2], \tag{61}$$

Notation	Domain	Description
K	$\mathbb{R}$	The abstracted input domain size
d	$\mathbb{R}$	The parameter dimension
θ	$\mathbb{R}^d$	The learned parameters
$\phi(x)$	$\mathbb{R}^d$	The input feature representations within f_θ
$\psi(x)$	$\mathbb{R}^d$	The input feature representations within g_θ
Φ	$\mathbb{R}^{K\times d}$	The compiled matrix of ϕ features
Ψ	$\mathbb{R}^{K\times d}$	The compiled matrix of ψ features
$\mathbf{D}_P$	$\mathbb{R}^{K\times K}$	The diagonal matrix of the P sampling probabilities
$\mathbf{Y}$	$\mathbb{R}^K$	The non-parametric labels
$\mathbf{P}^{\vert}$	$\mathbb{R}^{K\times K}$	The $P(x'\vert x)$ conditional distribution in matrix form
σ_M	$\mathbb{R}_0^+$	The spectral radius of Matrix M
$\mathbf{f}^*$	$\mathbb{R}^K$	The perfect f solution to Equation (1)
$\mathbf{g}^*$	$\mathbb{R}^K$	The perfect g solution to Equation (1)
Λ	$\mathbb{R}^{K\times K}$	The matrix relating the Φ and Ψ features
$\Pi\ \mathbf{H}$	$\mathbb{R}^K$	The projection operator to span(Φ)
$\mathcal{U}\ \mathbf{H}$	$\mathbb{R}^K$	The update operator to solve Equation (1)
$\Vert\mathbf{H}\Vert_P$	$\mathbb{R}_0^+$	The L2-norm weighted by the P distribution

Table 3. The notation used in the theoretical analyses of the delayed target method.

$$\mathcal{U}\ h := \mathbb{E}_{P(x'|\cdot)}\left[\int \Lambda_{x',x''} h(x'')\mathrm{d}x''\right] + y(\cdot). \tag{62}$$

Notice that

$$\mathcal{U}\ \Phi\theta = \mathbf{Y} + \mathbf{P}^{|}\Psi\theta. \tag{63}$$

Therefore, under these assumptions and notation, solving the original system defined by Equation (1) can be re-stated as finding the fixed point solution to the update operator:

$$\Phi\theta = \mathcal{U}\ \Phi\theta \quad \Longleftrightarrow \quad \forall x \in \mathcal{X} : f_\theta(x) = \mathbb{E}_{P(x'|x)}[g_\theta(x')] + y(x). \tag{64}$$

Table 3 summarizes this notation.

B.2. Theoretical Analysis

Here, we restate Theorem 4.1 as our main theoretical result. This theorem is a generalization of the ideas originally described in Tsitsiklis & Van Roy (1997).

Theorem 4.1. *Following the assumptions and notation in Section B.1, given learning rates that vanish neither too fast nor too slow, i.e.*

$$\sum_{t=0}^{\infty} \eta_t = \infty, \qquad \sum_{t=0}^{\infty} \eta_t^2 \leq \infty, \tag{65}$$

and using a small $\tau \to 0$ with $\lambda = 0$ and $N = 1$, the delayed target method is an instance of stochastic approximation (Robbins & Monro, 1951; Kiefer & Wolfowitz, 1952) and converges to the fixed point of the $\Pi\mathcal{U}$ composite operator in the following equation:

$$\Phi\theta^*_{DT} = \Pi\mathcal{U}\ \Phi\theta^*_{DT}. \tag{66}$$

Furthermore, assuming that f^* is the fixed point to the $\mathcal{U}$ operator in Equation (62), the approximation error for the delayed target method under these conditions is upper-bounded as

$$\mathbb{E}_{x\sim P}[(f_{\theta^*_{DT}}(x) - f^*(x))^2] \le \frac{1}{1-\sigma_{\mathbf{P}^{|}\Lambda}}\mathbb{E}_{x\sim P}\big[(\Pi\ f^*(x) - f^*(x))^2\big]. \tag{67}$$

Corollary B.1. *Under the conditions stated earlier and a realizability assumption over the function approximation class, that is, by having the perfect solution be a member of the function class:*

$$\exists\theta\ \forall x : f^*(x) = \phi(x)^{\mathrm{T}}\theta, \tag{68}$$

delayed targeting has no approximation error. That is, both sides of Inequality (67) are zero under a realizability assumption.

Moreover, due to the stochastic approximation properties, to reach an optimization error of ϵ, the total computational cost for the delayed target method can be $O(d/\epsilon)$, whereas the standard training with N samples can take $O(Nd\log(1/\epsilon))$ to achieve the same goal. Section B.3 will discuss this computational complexity analysis in more detail.

Proof. Given $d < K$ and the Moore-Penrose pseudo-inverse, we have

$$\Pi\ \mathbf{H} = \Phi(\Phi^{\mathrm{T}}\mathbf{D}_P\Phi)^{-1}\Phi^{\mathrm{T}}\mathbf{D}_P\mathbf{H}. \tag{69}$$

Consider the following Project-Update (PU) equation:

$$\Phi\theta = \Pi\mathcal{U}\ \Phi\theta. \tag{70}$$

The least-square solution to the PU equation, namely θ^*_{DT}, satisfies the following:

$$\theta^*_{\text{DT}} = (\Phi^{\mathrm{T}}\mathbf{D}_P\Phi)^{-1}\Phi^{\mathrm{T}}\mathbf{D}_P(\mathbf{Y} + \mathbf{P}^{|}\Lambda\Phi\theta^*_{\text{DT}}). \tag{71}$$

Therefore, we have

$$(\Phi^{\mathrm{T}}\mathbf{D}_P\Phi)\theta^*_{\text{DT}} = \Phi^{\mathrm{T}}(\mathbf{Y} + \mathbf{P}^{|}\Lambda\Phi\theta^*_{\text{DT}}). \tag{72}$$

Rearranging this will give us

$$\Phi^{\mathrm{T}}\mathbf{D}_P(I - \mathbf{P}^{|}\Lambda)\Phi\theta^*_{\text{DT}} = \Phi^{\mathrm{T}}\mathbf{D}_P\mathbf{Y}. \tag{73}$$

This is essentially to say θ^*_{DT} satisfies the $A\theta = b$ system, where

$$\mathbf{A} := \Phi^{\mathrm{T}}\mathbf{D}_P(I - \mathbf{P}^{|}\Lambda)\Phi, \tag{74}$$

$$\mathbf{b} := \Phi^{\mathrm{T}}\mathbf{D}_P\mathbf{Y}. \tag{75}$$

For a small τ, the Polyak averaging will produce almost identical θ and θ_{Target}:

$$\lim_{\tau\to 0}\theta^{(t)}_{\text{Target}} = \theta^{(t)}. \tag{76}$$

Under linear function approximation, the delayed target update at Iteration t simplifies to

$$\theta_{t+1} \leftarrow \theta_t - \eta_t \cdot (A_t\theta_t - b_t), \tag{77}$$

where we have

$$A_t := \phi(x)(\phi(x) - \psi(x')) \in \mathbb{R}^{d\times d}, \tag{78}$$

$$b_t := \phi(x)y(x) \in \mathbb{R}^d. \tag{79}$$

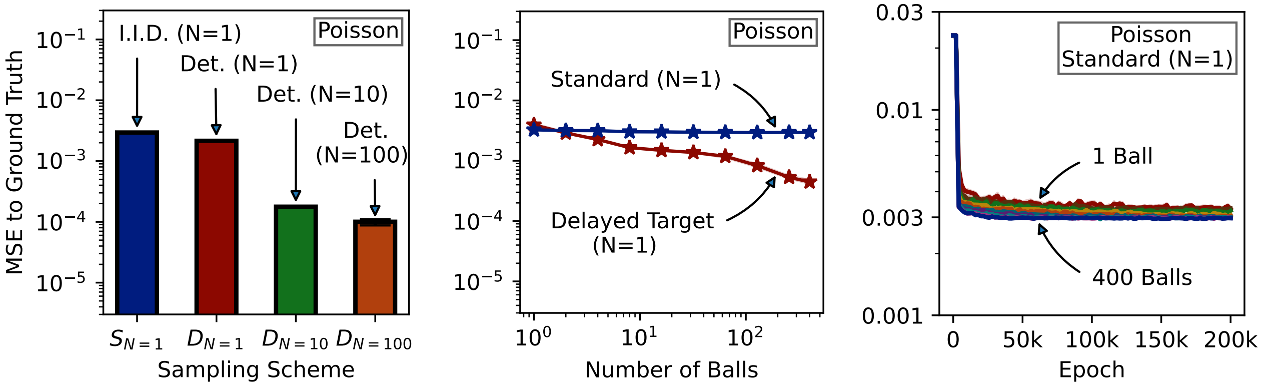

Figure 8. Ablation studies of the sampling hyper-parameters and settings in the 2D Poisson problem of Figure 1 in the main paper. *In the left plot*, we compare the deterministic and i.i.d. sampling on the standard trainings with various N. *In the middle plot*, the horizontal axis shows the number of balls sampled in each epoch. Both the standard and the delayed target methods are shown in this plot with $N = 1$. *The right plot* shows the training curves for the standard method with $N = 1$ target samples. Similar ablations for the Maxwell-Ampere and Smoluchowski problems are presented in Figure 9.

It is fairly straightforward to derive the expectation of A_t and b_t as

$$\mathbb{E}_{x\sim P, x'\sim P(\cdot|x)}[A_t] = \Phi^{\mathrm{T}}\mathbf{D}_P(I - \mathbf{P}^{|}\Lambda)\Phi = \mathbf{A}, \tag{80}$$

$$\mathbb{E}_{x\sim P, x'\sim P(\cdot|x)}[b_t] = \Phi^{\mathrm{T}}\mathbf{D}_P\mathbf{Y} = \mathbf{b}. \tag{81}$$

In other words, the delayed target method is an instance of stochastic approximation; the delayed target method is applying stochastic gradient descent to minimize $(\mathbf{A}\theta - \mathbf{b})^{\mathrm{T}}(\mathbf{A}\theta - \mathbf{b})$.

To upper-bound of the delayed target's approximation error, we can write

$$\begin{aligned}
\|\Phi\theta^*_{\mathrm{DT}} - \mathbf{f}^*\|_P &\le \|\Phi\theta^*_{\mathrm{DT}} - \Pi\ \mathbf{f}^*\|_P + \|\Pi\ \mathbf{f}^* - \mathbf{f}^*\|_P && (82)\\
&\le \|\Pi\mathcal{U}\ \Phi\theta^*_{\mathrm{DT}} - \Pi\ \mathbf{f}^*\|_P + \|\Pi\ \mathbf{f}^* - \mathbf{f}^*\|_P && (83)\\
&\le \|\mathcal{U}\ \Phi\theta^*_{\mathrm{DT}} - \mathbf{f}^*\|_P + \|\Pi\ \mathbf{f}^* - \mathbf{f}^*\|_P && (84)\\
&\le \|\mathcal{U}\ \Phi\theta^*_{\mathrm{DT}} - \mathcal{U}\ \mathbf{f}^*\|_P + \|\mathcal{U}\ \mathbf{f}^* - \mathbf{f}^*\|_P + \|\Pi\ \mathbf{f}^* - \mathbf{f}^*\|_P && (85)\\
&\le \sigma_{\mathbf{P}^{|}\Lambda}\|\Phi\theta^*_{\mathrm{DT}} - \mathbf{f}^*\|_P + \|\mathcal{U}\ \mathbf{f}^* - \mathbf{f}^*\|_P + \|\Pi\ \mathbf{f}^* - \mathbf{f}^*\|_P. && (86)
\end{aligned}$$

The first inequality in the chain is a triangle inequality. Inequality (83) holds since $\Phi\theta^*_{\mathrm{DT}}$ is a fixed point for $\Pi\mathcal{U}$. Inequality (84) holds because the $\Phi(\Phi^{\mathrm{T}}\mathbf{D}_P\Phi)^{-1}\Phi^{\mathrm{T}}\mathbf{D}_P$ matrix is idempotent and square. As a result, its eigenvalues can either be zero or one causing it to be a non-expansion. Inequality (85) is a triangle inequality applied after subtraction and addition of a $\mathcal{U}\ \mathbf{f}^*$ term. Finally, Inequality (86) holds because all eigenvalues of $\mathbf{P}^{|}\Lambda$ are upper-bounded by $\sigma_{\mathbf{P}^{|}\Lambda}$ in absolute value.

Assuming $\sigma_{\mathbf{P}^{|}\Lambda} < 1$, we can re-arrange Inequality (86) and write

$$\|\Phi\theta^*_{\mathrm{DT}} - \mathbf{f}^*\|_P \le \frac{1}{1 - \sigma_{\mathbf{P}^{|}\Lambda}}\big(\|\mathcal{U}\ \mathbf{f}^* - \mathbf{f}^*\|_P + \|\Pi\ \mathbf{f}^* - \mathbf{f}^*\|_P\big). \tag{87}$$

In other words, we have

$$\mathbb{E}_{x\sim P}[(f_{\theta^*_{\mathrm{DT}}}(x) - f^*(x))^2] \le \frac{1}{1 - \sigma_{\mathbf{P}^{|}\Lambda}}\mathbb{E}_{x\sim P}\big[(\Pi\ f^*(x) - f^*(x))^2 + (\mathcal{U}\ f^*(x) - f^*(x))^2\big]. \tag{88}$$

This essentially upper-bounds the MSE error to the ground truth by a $(1 - \sigma_{\mathbf{P}^{|}\Lambda})^{-1}$ coefficient times the projection and update errors of the true f^* to the function approximation class (i.e., the function approximation error). Given the assumption that f^* is the fixed point to the $\mathcal{U}$ operator, Inequality (88) reduces to the upper-bound stated in the theorem. □

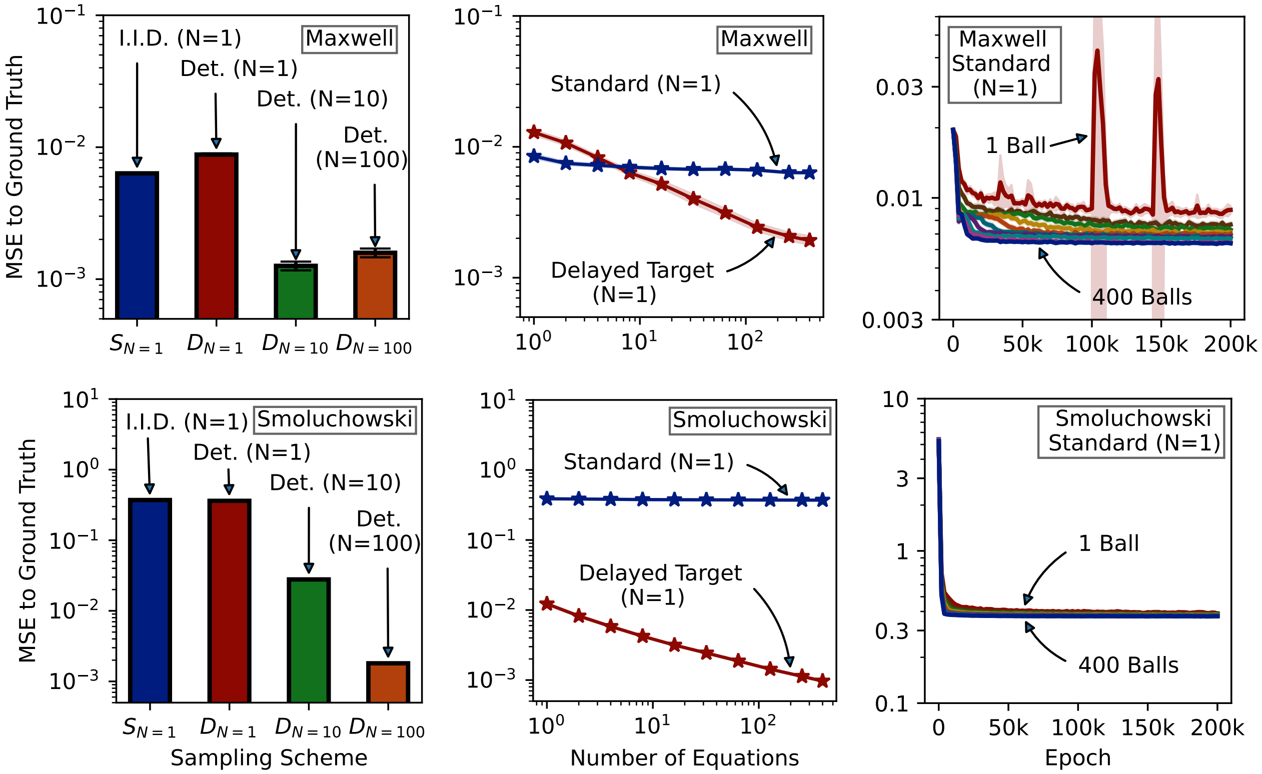

Figure 9. Ablation studies of the sampling hyper-parameters and settings in the Maxwell-Ampere and Smoluchowski problems. *The top panel* corresponds to the Maxwell-Ampere problem of Figure 5 in the main paper, and the bottom panel corresponds to the 2D Smoluchowski problem of Figure 6 in the main paper. *In the left column*, we compare the deterministic and i.i.d. sampling on the standard trainings with various N. *In the middle column*, the horizontal axis shows the number of balls or equations sampled in each epoch. Both the standard and the delayed target methods are shown in this plot with $N = 1$. *The right column* shows the training curves for the standard method with $N = 1$ target samples. Similar ablations for the Poisson problem are presented in Figure 8.

B.3. The Computational Complexity of the Delayed Target Method

For the sake of simplicity, assume that the delayed target and standard methods are executed over the abstracted space with maximal x batch-sizes. In other words, consider the case where at each iteration, the gradients for all $x \in \mathcal{X}$ are estimated using either the delayed target or standard methods, and then the parameters are updated with the average gradient estimates of all $x \in \mathcal{X}$. We mainly make this assumption to strip the irrelevant $x \sim P$ sampling stochasticity from the standard method and make it deterministic. As a result, the standard method can be expressed as a gradient descent optimization instance. However, we still assume the $x' \sim P(\cdot|x)$ sampling stochasticity to remain within the delayed target method.

As we discussed, the delayed target method is an instance of stochastic approximation:

1. The cost of each iteration for the standard training method is $O(Nd)$. However, the cost for each iteration of the delayed target method is $O(d)$ under the conditions of Theorem 4.1.
2. To achieve an optimization error of ϵ in its respective problem, the standard training method needs $O(\log(1/\epsilon))$ iterations, since it is an instant of the gradient descent algorithm. However, the delayed target method needs $O(1/\epsilon)$ iterations to reach the same optimization error, since it is an instance of the stochastic gradient descent algorithm.

Therefore, the total computational cost for the delayed target method to reach an optimization error of ϵ is $O(d/\epsilon)$. However, the standard training method needs $O(Nd\log(1/\epsilon))$ to achieve the same goal.

Notice that the gradient averaging assumption over all $x \sim P$ is not overly restrictive. Both methods can identically use a smaller x batch-size in practice, and the computational complexity insights regarding the x' stochasticity remain applicable. We only introduced this assumption to make the computational complexity analysis easier to express.

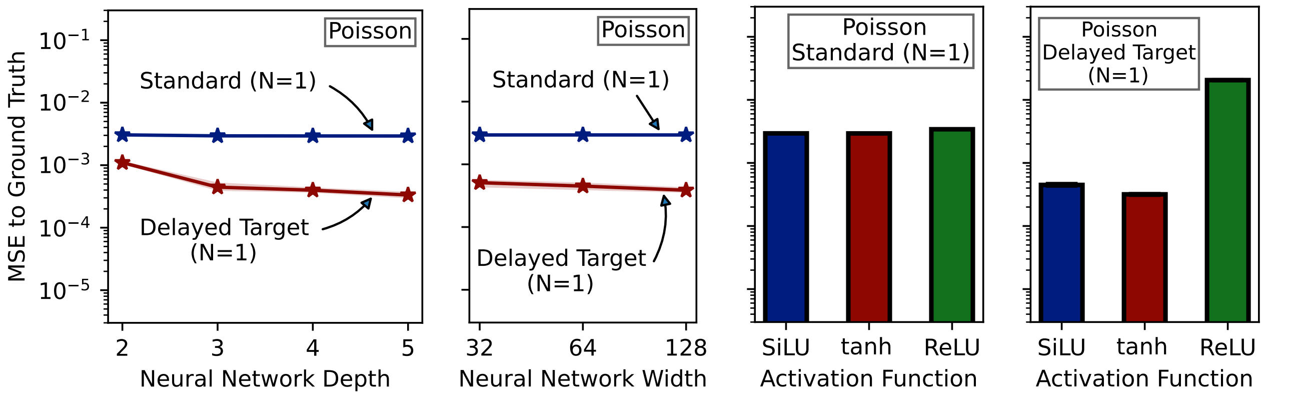

Figure 10. Ablating the function approximation class attributes on the 2D Poisson problem of Figure 1 in the main paper. A multi-layer perceptron was used in all of our experiments. *The left and right line plots* show the effect of the neural network's depth and width, respectively, on each of the standard and delayed target methods. *The left and right bar plots* demonstrate the effect of the neural activation function on the standard and the delayed target methods, respectively. These results indicate that the function approximation class can have a more substantial impact on the delayed target method than the standard trainings. Similar ablations for the Maxwell-Ampere and Smoluchowski problems are presented in Figure 11.

C. Ablation Studies

Here, we examine the effect of different design choices and hyper-parameters with various ablation studies.

C.1. Surface Point Sampling Scheme Ablations

Figure 8 compares the deterministic vs. i.i.d. sampling schemes and the effect of various mini-batch sizes (i.e., the number of volumes sampled in each epoch) on the Poisson problem of Figure 1 in the main paper. Figure 9 shows similar ablations for the Maxwell-Ampere and Smoluchowski problems. The results suggest that the deterministic sampling scheme can train successfully with large N, however, it may not remedy the biased solution problem with the standard training at small N values. Furthermore, the results indicate that the number of volumes in each epoch has minimal to no effect on the standard training method, which indicates that such a parallelization is not the bottleneck for the standard training method. To make this clear, we showed the training curves for the standard method, and they indicate similar trends and performance across a large range (1 to 400) of batch-size values for the standard method.

On the other hand, the performance of the delayed target method improves upon using a larger batch size, which possibly indicates that this problem has a high objective estimation variance. The ability to trade small-quantity high-quality data (i.e., large N with small mini-batch sizes) with large-quantity low-quality data (i.e., small N with larger mini-batch sizes) is an advantage of the delayed target method relative to standard trainings.

C.2. Function Approximation Ablations

Figure 10 compares the effect of the neural architecture parameters on the performance of the standard training vs. the delayed target method on the 2D Poisson problem of Figure 1 in the main paper. Figure 11 shows similar ablations for the Maxwell-Ampere and Smoluchowski problems. The standard training exhibits a steady performance across all neural network depths, widths, and activation functions. However, the performance of the delayed target method seems to be enhancing with deeper and wider networks. The effect of the neural activation functions is more pronounced than the depth and width of the network. In particular, the ReLU activation performs substantially worse than the $\tanh$ or SiLU activations, and the SiLU or $\tanh$ activations seem to work similarly.

To shed some further light on the training behavior of the delayed target method, we show the training curves for different neural hyper-parameters in Figures 12 and 13. The results indicate that the ReLU activation prevents the delayed target method from improving during the entire training. On the other hand, the SiLU activation yields better initial improvements but struggles to maintain this trend consistently in the Poisson and Maxwell-Ampere problems. Based on this, we speculate that some activation functions (e.g., ReLU) may induce poor local optima in the optimization landscape of the delayed target method, which may be difficult to run away from.

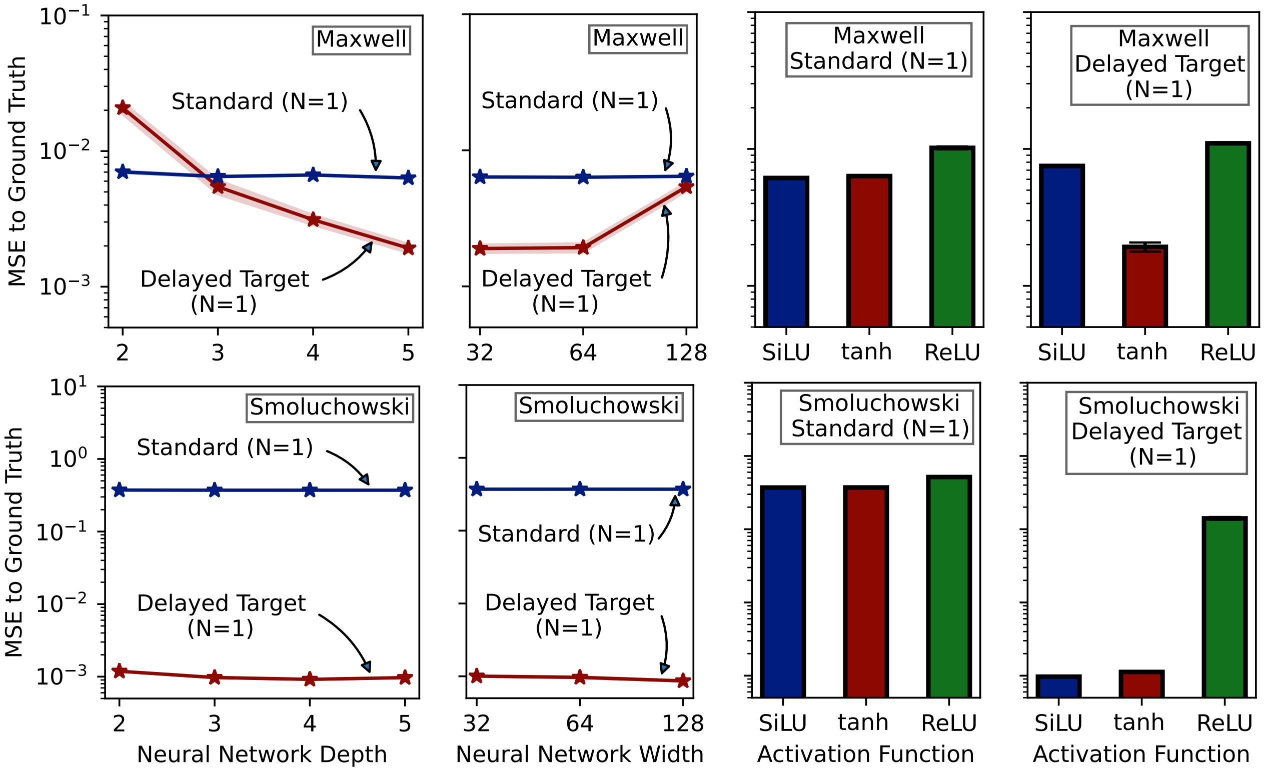

Figure 11. Ablating the function approximation class attributes on the Maxwell-Ampere and Smoluchowski problems. A multi-layer perceptron was used in all of our experiments. *The top panel* corresponds to the Maxwell-Ampere problem of Figure 5 in the main paper, and the bottom panel corresponds to the 2D Smoluchowski problem of Figure 6 in the main paper. *The left and right line plots* show the effect of the neural network's depth and width, respectively, on each of the standard and delayed target methods. *The left and right bar plots* demonstrate the effect of the neural activation function on the standard and the delayed target methods, respectively. These results indicate that the function approximation class can have a more substantial impact on the delayed target method than the standard trainings. Similar ablations for the Poisson problem are presented in Figure 10.

Our neural depth analysis in Figure 12 indicates that deeper networks can yield quicker improvements in performance. However, such improvements are difficult to maintain stably over the entire course of training. In particular, the 2-layer training yields worse performance than the deeper networks, but maintains a monotonic improvement throughout the training, unlike the other methods. Of course, such behavior may be closely tied together with the activation function used for function approximation, as we discussed earlier. On the other hand, deeper networks in the Maxwell-Ampere problem produce better results consistently in Figure 13. In the Smoluchowski problem, the depth of the network makes little to no difference in the performance. These examples demonstrate a variety of different behaviors for the role of neural depth in the delayed target method. A better understanding of this pathology with respect to the problem conditions and the rest of the hyper-parameters is an important topic for future research.

We also show the effect of network width on the performance of the delayed target method in Figures 12 and 13. In the Poisson problem, wider networks are more likely to provide better initial improvements. In the Maxwell-Ampere problem, the widest network produces poor performance. Finally, in the Smoluchowski problem, there is no substantial difference in performance between different network widths. That being said, as the networks are trained for longer, the differences in performance between different network widths shrink, and narrower networks tend to show similar final performances as the wider networks (assuming a stable training behavior).

All in all, our results indicate that the choice of the neural function approximation class, particularly with varying activation and depths, can have a notable impact on the performance of the delayed target method. We speculate that this is due to the incomplete gradients used during the optimization process of the delayed target method. The effect of incomplete function approximation on bootstrapping methods has been studied frequently, both in theory and practice, in other contexts such as the Fitted Q-Iteration (FQI) and Q-Learning methods within reinforcement learning. This is in contrast to the standard training methods, which seem quite robust to function approximation artifacts at the expense of solving an excess-variance diluted optimization problem.

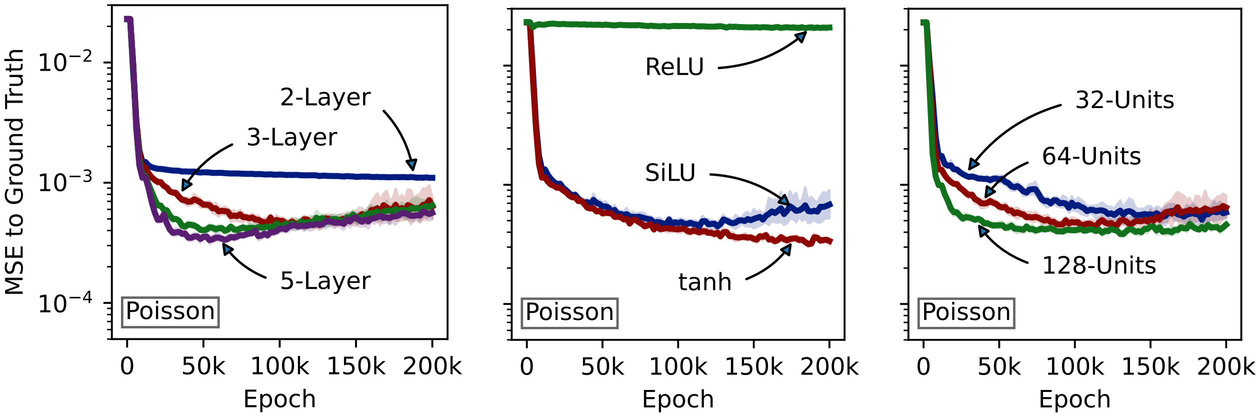

Figure 12. A closer look at the training curves for the delayed target method with different neural network hyper-parameters on the 2D Poisson problem of Figure 1 in the main paper. *The left, middle, and right plots* show the training curves for various neural depths, activations, and widths, respectively. Similar ablations for the Maxwell-Ampere and Smoluchowski problems are presented in Figure 13.

C.3. Integration Volume Sampling Ablations

For our integration volumes, we randomly sampled balls of varying radii and centers. The distribution of the sampled radii and centers could impact the performance of different methods. Figure 14 studies such effects on both the standard and the delayed target methods in the 2D Poisson problem of Figure 1 and the Maxwell-Ampere problem of Figure 5 in the main paper. In short, we find that the delayed target method is robust to such sampling variations; an ideal method should find the same optimal solution with little regard to the integration volume distribution. On the other hand, our results indicate that the standard training performance tends to be sensitive to the integration volume distributions. This may be because the standard trainings need to minimize two loss terms; the optimal balance between the desired loss function $\mathcal{L}_\theta(x)$ and the excess variance $\mathbb{V}_{P(x'|x)}[g_\theta(x')]$ in Equation (26) may be sensitive to the distribution of x itself.

C.4. Poisson charge placement ablations

The charge locations in the 2D Poisson problem may impact the results of our methods. For this, we compare the standard and the delayed target method over a wide variety of charge distributions. Figure 15 summarizes these results. Here, the three fixed charge locations shown in Figure 1 of the main paper are shown as a baseline. We also show various problems where the charge locations were picked uniformly or normally in an i.i.d. manner. The performance trends seem to be quite consistent for each method, and the fixed charge locations seem to represent a wide range of such problems and datasets.

C.5. Robustness to the Initial Conditions

Our method is extremely robust to the neural network initializations as shown by the small confidence intervals in our results. In addition, physics-informed networks can readily handle different PDE initial conditions. The delayed target method is less sensitive to the weight placed on the initial condition loss term since it can effectively eliminate the excess variance term. This is in contrast to the standard training method where the initial condition enforcement may be negatively influenced by the excess variance term.

C.6. Sampling Quadrature and QMC Integration Points

Obtaining uniform samples on the surface of the integration volumes is a necessary step to estimate the divergence theorem integrals. Deterministic methods, such as numerical quadrature and QMC, can be sensitive to the specific choice of these points as they define a constant arrangement of surface points for the entirety of the training. On the other hand, stochastic methods, such as the standard, the double-sampling, and the delayed target methods, can be less sensitive to this problem as they sample the integration points in an i.i.d. manner and any point on the integration surfaces have the same non-zero probability of appearing in the integral estimates at each iteration.

For instance, consider a 2D Poisson problem. Uniform samples on the surface of a 2D unit ball (i.e., the unit circle) can be obtained in one of two ways. One approach is to sample 2-dimensional Gaussian random variables and normalize them

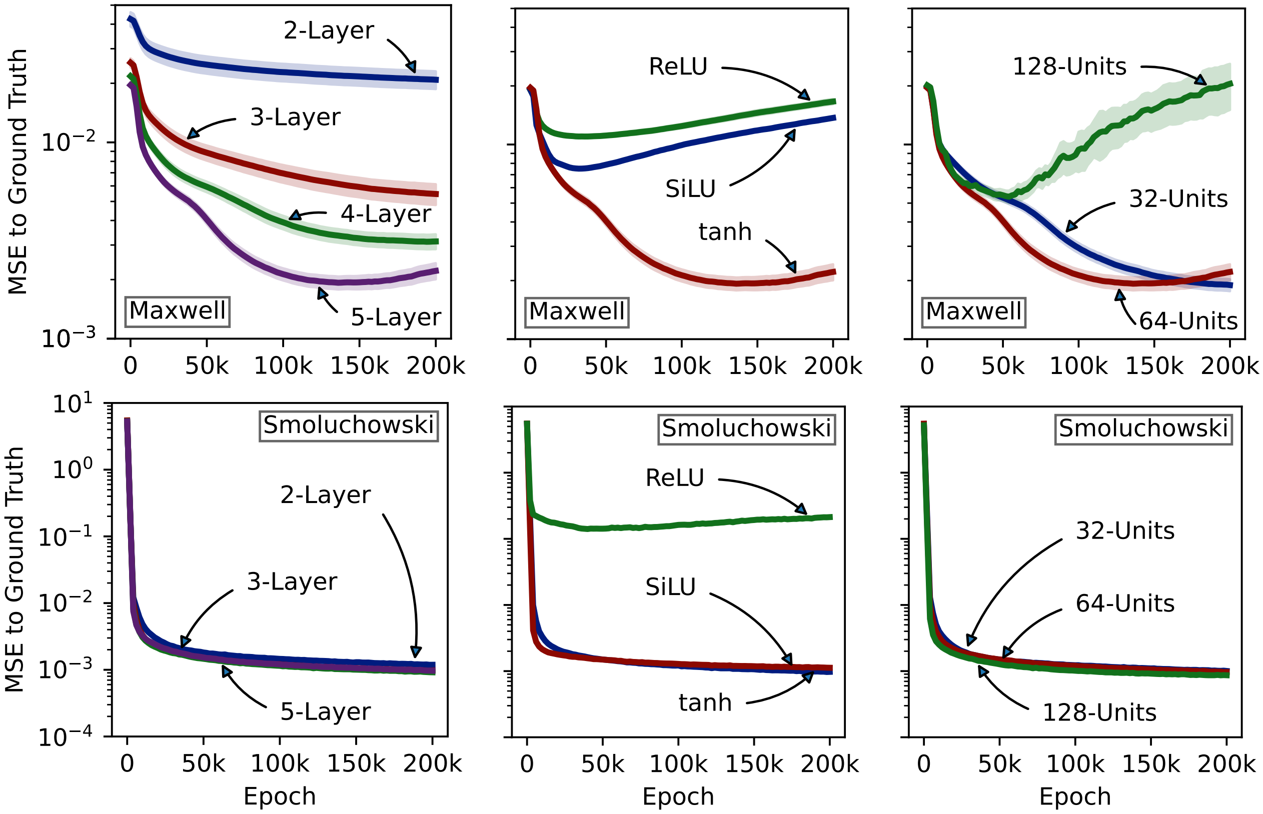

Figure 13. A closer look at the training curves for the delayed target method with different neural network hyper-parameters on the Maxwell-Ampere and Smoluchowski problems. *The top panel* corresponds to the Maxwell-Ampere problem of Figure 5 in the main paper, and the bottom panel corresponds to the 2D Smoluchowski problem of Figure 6 in the main paper. *The left, middle, and right columns* show the training curves for various neural depths, activations, and widths, respectively. Similar ablations for the Poisson problem are presented in Figure 12.

so that they fall on the unit circle. This process defines a 2-dimensional integral since the normal random variables were sampled in the 2-dimensional space. Alternatively, we could sample uniform variables in the 1-dimensional space, and apply an appropriate transformation so that they cover the unit circle. This approach defines a 1-dimensional integral, instead.

This distinction can be generalized to higher dimensional problems as well; to solve a d-dimensional Poisson problem, the Gaussian sampling and normalization approach defines a d-dimensional integral for the divergence theorem, while sampling from the $(d-1)$-dimensional cube and transforming the points to the surface of the d-dimensional unit ball defines a $(d-1)$-dimensional integral for the divergence theorem. Although both estimators are statistically consistent (i.e., yield the same integrals with infinite samples), under finite sample sizes, they define different integration point distributions on the integration surfaces for the deterministic methods.

Figure 18 studies the effect of these sampling procedures on the deterministic methods in a 2D Poisson problem from Figure 4 of the main paper. Sampling points from the $(d-1)$-dimensional area and transforming them to the surface of a d-dimensional ball seems to work best for both the QMC and numerical quadrature methods. However, numerical quadrature seems to be particularly sensitive to this choice; our results suggest that the specific choice of the d- or $(d-1)$-dimensional integral estimators can significantly impact the performance of numerical quadrature.

Based on these results, we sampled points from the $(d-1)$-dimensional cube and transformed the points to the surface of the d-dimensional ball for the QMC and quadrature methods. This was done to present these methods in the best light.

C.7. Joint Target Smoothing and Regularization

Both the target smoothing (τ) and regularization (λ) weights have a role in regulating the delayed target updates. In particular, target smoothings regulate the target model updates. On the other hand, the target regularization controls the main model

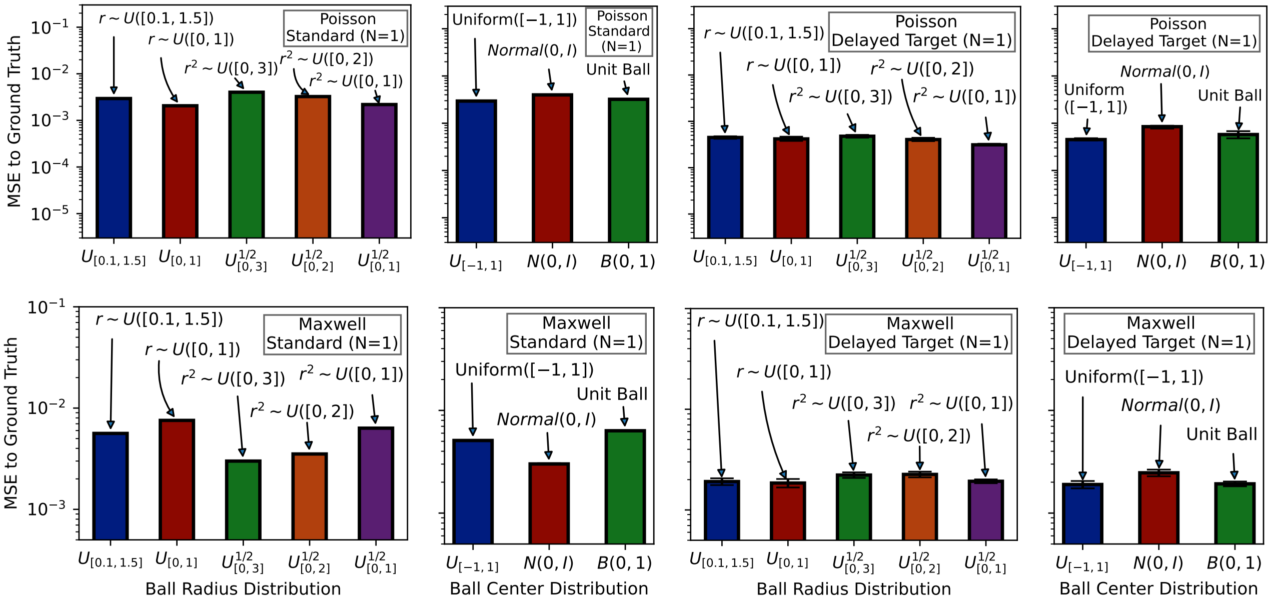

Figure 14. Ablating the distribution of ball centers and radii for both of the standard and the delayed target methods, both with $N = 1$. *The top panel* corresponds to the 2D Poisson problem of Figure 1 in the main paper, and the bottom panel corresponds to the Maxwell-Ampere problem of Figure 5 in the main paper. *The two left columns* correspond to the standard method, and the *the two right columns* correspond to the delayed target method. In each quarter, *the left plot* shows the effect of the ball radius distribution, and *the right plot* shows the effect of the ball center distribution. In the radius ablations, $r \sim U([a, b])$ means the ball radius was sampled from a uniform distribution over $[a, b]$. Also, $r^2 \sim U([a, b])$ means that the radius was the square root of a uniform random variable between a and b. The ball centers were randomly picked either (1) uniformly over a square with the $[-1, -1], [-1, 1], [1, 1], [1, -1]$ vertices, (2) normally, or (3) uniformly within the unit ball.

updates and prevents the main parameters from diverging from the vicinity of the target parameters. One may wonder how these two factors are practically different when regulating the parameter updates.

To understand the joint impact of these regulation factors, Figure 19 shows the delayed target training curves on a grid of these hyper-parameters. Since this is a singular problem and we used $M = 10^3$ with $N = 1$, this is a relatively challenging problem for the delayed target method to solve. The results indicate that both the target smoothing and regularization factors play a role in controlling the convergence of the delayed target method. The λ target regularization can stabilize the training curves in such challenging setups. On the other hand, tuning the target smoothing τ can improve the peak performance of the method. Together, these two factors can regulate the delayed target updates effectively.

C.8. Further Delayed Target Ablations

Three main hyper-parameters are involved in the definition of the delayed target method: (1) the target smoothing τ, (2) the target regularization weight λ, and (3) the target weight M described in Equation (36). Figures 16 and 17 study the effect of each of these hyper-parameters on the performance of the delayed target method in the Poisson, Maxwell-Ampere, and Smoluchowski problems.

Our results indicate that choosing a proper target smoothing can improve the performance of the delayed target method. In particular, neither a significantly small nor a substantially large τ can yield optimal training. Small τ values cause the training target to evolve rapidly. This may accelerate the training initially, but it can negatively impact the final performance of the method as we show in Figure 16. On the other hand, too large values of τ can cause the target network to lag behind the main solution, thus bottlenecking the training. The optimal τ in this problem defines a smoothing window of size $1/(1 - \tau) = 1000$ in the Poisson problem, which seems small enough for a training duration of $200k$ epochs. On the other hand, the optimal smoothing window is much smaller in the Maxwell-Ampere problem, and the delayed target method benefits from more frequent changes to the target parameters in this problem.

Next, we studied the effect of target regularization weight λ in Algorithm 1 of the main paper. A small target weight causes this method to diverge in this particular problem, as we've shown in Figure 3 of the main paper. On the other hand, a

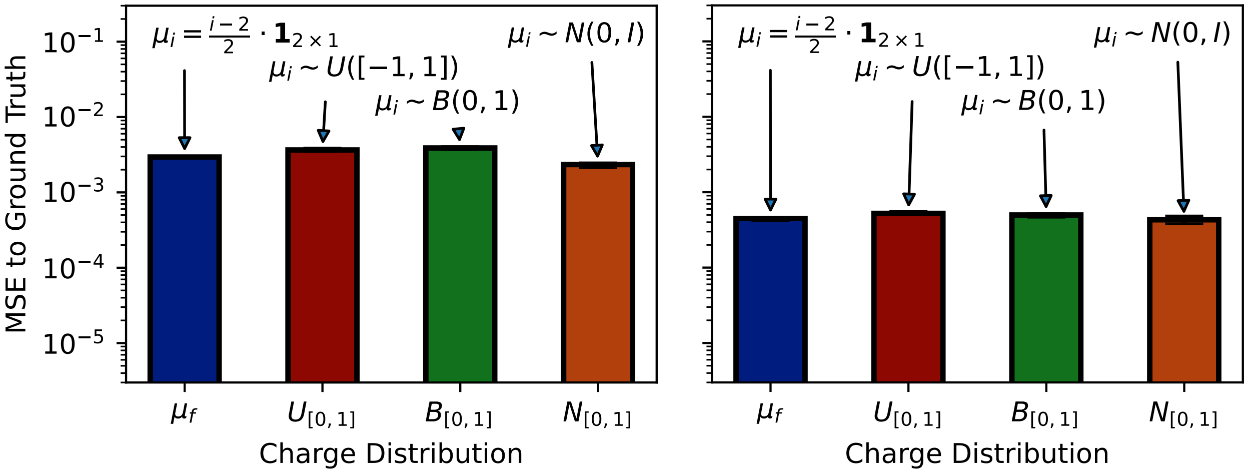

Figure 15. Ablating the location distribution of the three charges on the 2D Poisson problem of Figure 1 in the main paper. *The left plot* shows the results for the standard training method, while the *the right plot* corresponds to the delayed target method, all with $N = 1$. The blue bar represents fixing the charge locations at the $[0, 0]$, $[-0.5, -0.5]$, and $[0.5, 0.5]$ coordinates. We also show the results for picking the charge locations in an i.i.d. manner (1) uniformly between $[-1, -1]$ and $[1, 1]$ (denoted as $U([-1, 1])$, (2) uniformly over the unit ball (denoted as $B(0, 1)$), and (3) normally (denoted as $N(0, I)$).

regularization weight too large can slow down the training, as the main model remains too constrained to the target model during training.

We also show the effect of various target weight M values in this problem. Ideally, $M \to \infty$ to make our approximations more accurate. A small target weight can effectively cause the method to seek biased solutions. On the other hand, setting M too large may be impractical and instead cause the loss estimator's variance to explode as discussed in Equation (36). For this, M must be set in conjunction with the λ hyper-parameter in such challenging problems.

Illustration of failure modes: The delayed target method is more temperamental than the standard training; the set of delayed-target hyper-parameters, such as λ and τ, can have a significant impact on the solution quality. With poor hyper-parameters, the delayed target may poorly track the main solution, and the method can certainly diverge under an inappropriate set of hyper-parameters as we show in Figure 3 of the main paper. Figure 16 also details the impact of the hyper-parameters related to the delayed target method. Furthermore, Equation (25) indicates that the excess-variance problem can be less severe when the underlying true solution is smooth in g (i.e., when the optimal solution θ^* has a small $\mathbb{V}_{P(x'|x)}[g_\theta(x')]$ variance). Therefore, when the g_{θ^*} landscape is nearly flat, we expect the standard training to perform as well as the proposed method.

C.9. Delayed Target Sample Size Scaling

Two sets of parameters θ and θ^{Target} participate in forming the training loss for the delayed target method. With $N = 1$, half of the evaluated terms back-propagate the gradients; only the terms parameterized by θ can contribute to the gradient. With larger N, the target values become less noisy, but only a single term can still back-propagate the gradient. This is in contrast to all the other methods (e.g., the standard, the double-sampling, the deterministic approaches), where all of the evaluated points backpropagate gradients to the main set of parameters. One may wonder if this is the most efficient use of the evaluated terms for gradient estimation.

To account for this, instead of solving for

$$f_\theta(x) = \frac{1}{N}\sum_{i=1}^{N} g_\theta(x'_i) + y(x), \tag{89}$$

in Equation (22), we can reformulate the main objective to evaluate N' points using the main model as

$$\frac{1}{N'}\sum_{j=1}^{N'} f_\theta(x_j) = \frac{1}{N}\sum_{i=1}^{N} g_\theta(x'_i) + y(x_1, \cdots, x_{N'}). \tag{90}$$

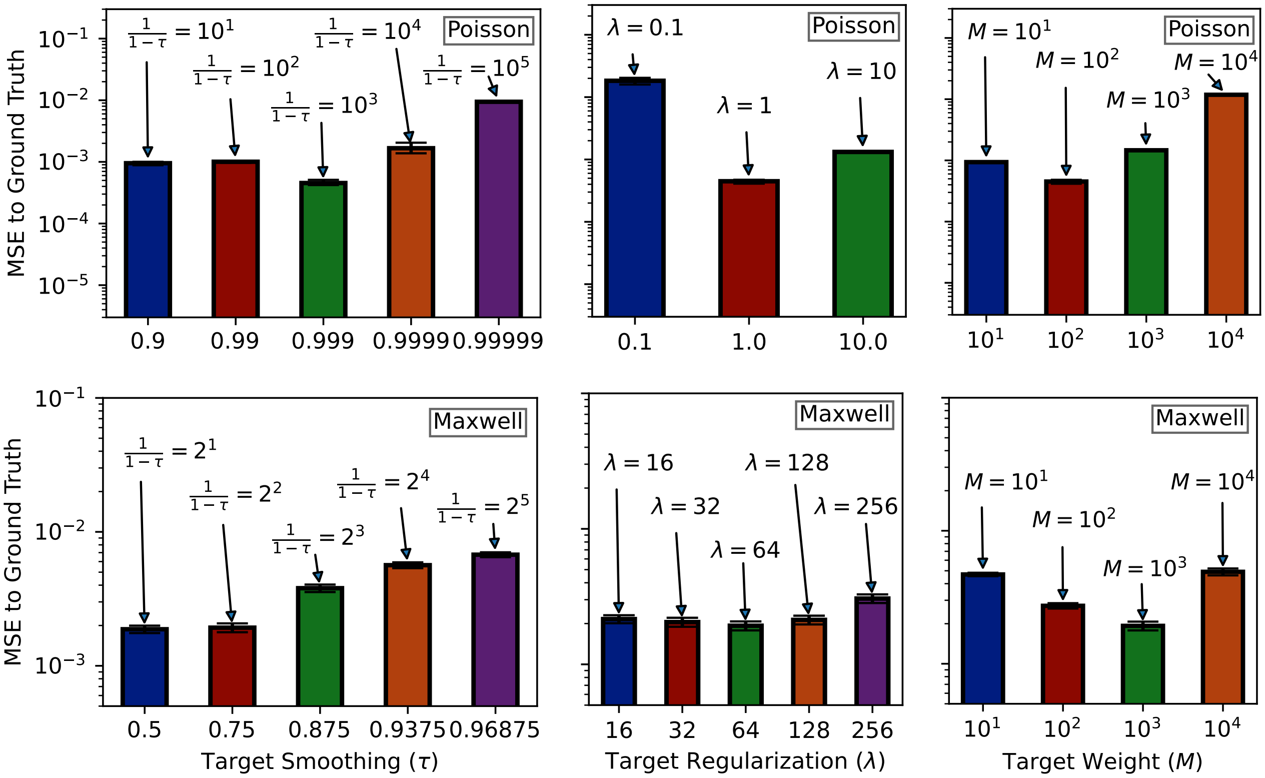

Figure 16. Ablating the effect of the delayed target method parameters on the Poisson and Maxwell-Ampere problems. *The top panel* corresponds to the 2D Poisson problem of Figure 1 in the main paper, and the bottom panel corresponds to the Maxwell-Ampere problem of Figure 5 in the main paper. *The left column* shows the effect of the target smoothing parameter τ in Algorithm 1 of the main paper. *The middle column* shows the effect of the target regularization parameter λ in Algorithm 1 of the main paper. *The right column* shows the effect of the target weight factor M in Equation (36). Similar ablations for the Smoluchowski problem are presented in Figure 17.

This is particularly straightforward in Examples 2.1 and 2.2, where an arbitrary number of points can be assigned to either side of the equation. In general, a portion of the $\frac{1}{N}\sum_{i=1}^{N} g_\theta(x'_i)$ terms in Equation (1) can always join the f_θ terms on the other side of the equation to form a training loss. This generalization allows the delayed target method to back-propagate the main parameter gradients using more points.

Of course, the introduction of the N' hyper-parameter begs more questions about properly setting up the delayed target method. For instance, one may wonder whether the specific choice of N and N' could impact this method when the total number of samples $N + N'$ is controlled. For instance, one scaling strategy could be to maintain $N' = 1$ while increasing N. Similarly, one could keep $N = 1$ while increasing N'. Another option is to scale both N and N' equally.

Figure 20 shows the effect of increasing the N and N' sample sizes within the delayed target method. In particular, we see that having $N' = 5$ and $N = 1$ yields a better performance than having $N' = 1$ with $N = 5$. This is consistent with the theory that higher N' values may lead to better gradient estimates to update the main model. However, having $N = N' = 3$ seems to yield better results than both of the aforementioned approaches. This means that balancing the main and target terms may be the best decision when generalizing the delayed target method to larger sample sizes.

Based on these results, we used $N = 5$ and $N' = 6$ to obtain the $N = 10$ curve in Figure 4 of the main paper. We abused the notation when labeling this curve to simplify the comparison between different methods (since the $N = 10$ and $N' = 1$ configuration has the same $N + N'$ total value). Similarly, we used $N = 50$ and $N' = 51$ for the delayed target curve labeled $N = 100$ in the same figure.

C.10. Learning Rate Ablation

Figure 21 shows the effect of the optimization learning rate on the standard method with $N = 1$ in the Poisson, Maxwell-Ampere, and Smoluchowski problems. The poor performance of the standard method is consistent for a wide range of

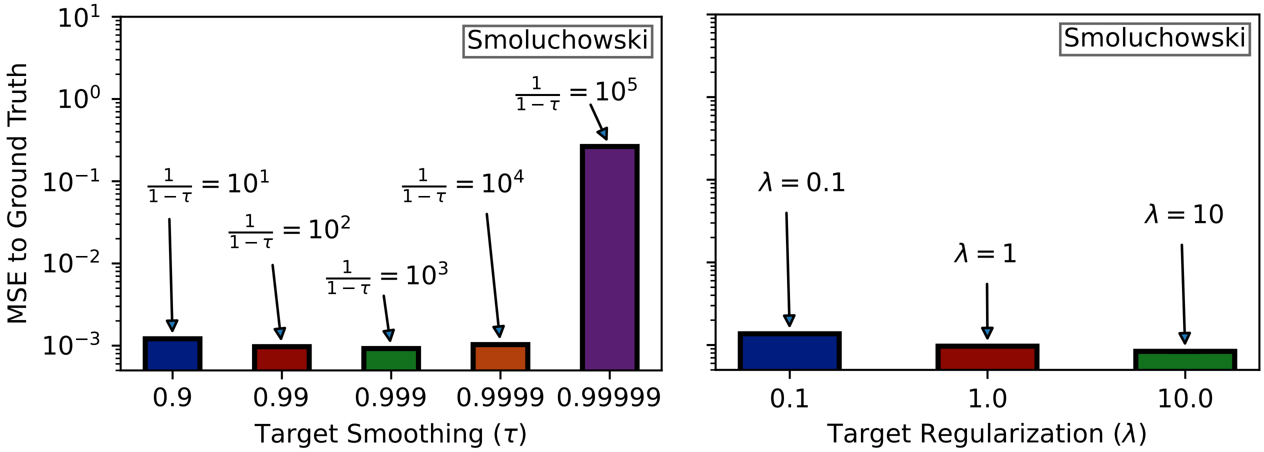

Figure 17. Ablating the effect of the delayed target method parameters on the 2D Smoluchowski problem of Figure 6 in the main paper. *The left column* shows the effect of the target smoothing parameter τ in Algorithm 1 of the main paper. *The right column* shows the effect of the target regularization parameter λ in Algorithm 1 of the main paper. Similar ablations for the Poisson and Maxwell-Ampere problems are presented in Figure 16.

optimization learning rates. These results suggest that decaying the learning rate is not an effective solution to address the biased nature of the standard method's training objective.

D. Implementation Details

In this section, we note the implementation details for each of the discussed problems. Sections D.1, D.2, and D.3 describe the implementation details for the Poisson, Maxwell-Ampere, and Smoluchowski problems, respectively. Sections D.4 and D.5 derive the analytical solutions for the Poisson and Maxwell-Ampere equations. Section D.6 discusses the random effect matching process used throughout the experiments. Finally, Section D.7 details the training hyper-parameters used in our numerical experiments.

D.1. The Main Poisson Problems

To solve this system in the integrated form, the standard method consists of fitting a neural model to the following loss:

$$\hat{\mathcal{L}} = \mathbb{E}\bigg[\Big(A_d^r \cdot \frac{1}{N}\sum_{i=1}^{N} E_\theta(x^{(i)}) \cdot \hat{n}(x_i) - y_\Omega\Big)^2\bigg], \tag{91}$$

where $A_d^r := \int_{\partial\Omega_r} 1 \, \mathrm{d}S$ is the surface area of a d-dimensional ball with the r radius, and the label is $y_\Omega := \iint_\Omega \nabla \cdot E \mathrm{d}V$. The x_i samples follow the $\mathrm{Unif}\big(\partial\Omega_r\big)$ distribution. The sampling intensity for volume Ω defines the test constraint weights.

In Figures 1, 2, and 3 of the main paper, we consider a Poisson problem with $d = 2$ dimensions and Dirac-delta charges. We place three unit charges at $[0, 0]$, $[-0.5, -0.5]$, and $[0.5, 0.5]$ coordinates. For this setup, computing y_Ω is as simple as summing the charges residing within the volume Ω. The integration volumes are defined as random spheres. The center coordinates and the radius of the spheres are sampled uniformly in the $[-1, 1]$ and $[0.1, 1.5]$ intervals, respectively. We train all models for 200,000 epochs, where each epoch samples 1000 points in total. We also study higher-dimensional problems with $d \in [2, 10]$ with a single charge at the origin in Figure 4 of the main paper.

D.2. The Maxwell Problem with a Rectangular Current Circuit

Our second example looks at finding the magnetic potentials and fields in a closed circuit with a constant current. This defines a singular J current density profile. We consider a rectangular closed circuit in the 3D space with the $[\frac{1}{\sqrt{3}}, \frac{-1}{\sqrt{3}}, \frac{-1}{\sqrt{3}}]$, $[\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}]$, $[\frac{-1}{\sqrt{3}}, \frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}]$, and $[\frac{-1}{\sqrt{3}}, \frac{-1}{\sqrt{3}}, \frac{-1}{\sqrt{3}}]$ vertices. The training volumes were defined as random circles, where the center coordinates and the surface normals were sampled from the unit ball, and the squared radii were sampled uniformly in the $[0.0, 1.0]$ interval.

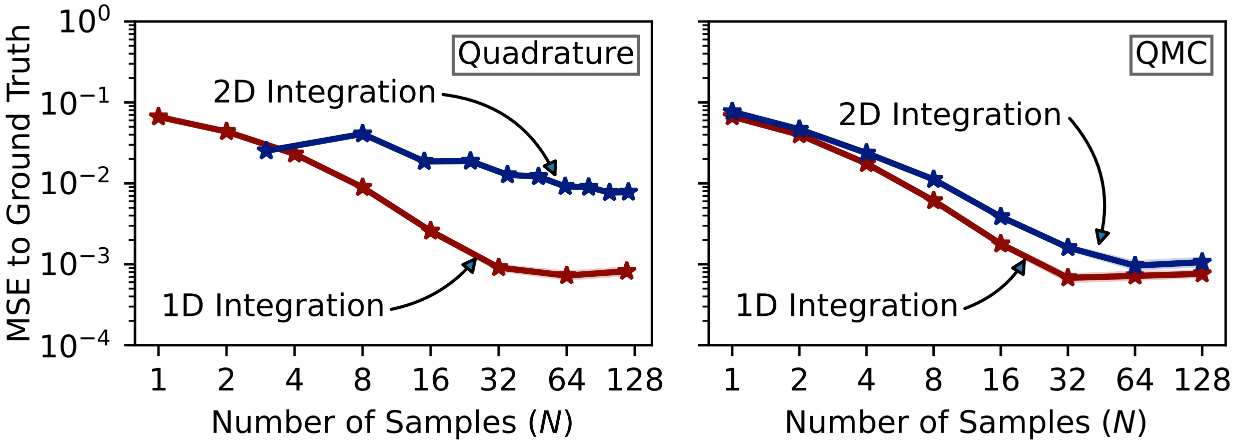

Figure 18. Demonstrating the effect of the specific sampling procedures on the quadrature and QMC methods. The *left and the right plots* correspond to Gaussian quadrature and QMC methods, respectively. All runs were performed on a 2D Poisson problem from Figure 4 of the main paper. No sparse grids were used here. Each curve denotes a different sampling procedure for obtaining the surface points in the divergence theorem. 2D integrations were defined by sampling normal random variables and normalizing them to fall on the unit circle. 1D integrations directly sampled uniform points on the unit circle. The latter approach yields the best results, and numerical quadrature seems to be particularly sensitive to this choice.

D.3. Smoluchowski Coagulation Problem

To simulate particle evolution dynamics, we consider a Smoluchowski coagulation problem where particles evolve from an initial density. We considered the x and x' particle sizes to be in the $[0, 1]$ unit interval, and the simulation time to be in the $[0, 1]$ unit interval as well. We designed the $K(x, x')$ coagulation kernel to induce non-trivial solutions in our unit solution intervals. Specifically, we defined $K(x, x') = 1.23 \times (\min(1.14, \sqrt{x} + \sqrt{x'}))^3$. To find a reference solution, we performed Euler integration using exact time derivatives on a large grid size. The grid time derivatives were computed by evaluating the full summations in the Smoluchowski coagulation equation.

D.4. The Analytical Solution to the Poisson Problem

Consider the d-dimensional space $\mathbb{R}^d$ and the following charge:

$$\rho_x = \delta^d(x). \tag{92}$$

For $d \neq 2$, the analytical solution to the $E = \nabla U$ and $\rho = \nabla \cdot E$ system of Equations (2) and (3) can be derived as

$$U(x) = \frac{\Gamma(d/2)}{2 \cdot \pi^{d/2} \cdot (2-d)} \|x\|^{2-d}, \tag{93}$$

$$E(x) = \frac{\Gamma(d/2)}{2 \cdot \pi^{d/2} \cdot \|x\|^d} x. \tag{94}$$

For $d = 2$, E_x stays the same but we have $U(x) = \frac{1}{2\pi} \ln(\|x\|)$. To solve this system using the divergence theorem in Equation (4), we can turn the integrals into scaled expectations. To find the appropriate scale, the $d-1$-dimensional surface area of a d-dimensional sphere with a radius of r can be described as

$$A_d^r := \int_{\partial\Omega_r} 1 \, \mathrm{d}S = \frac{2 \cdot \pi^{d/2}}{\Gamma(d/2)} \cdot r^{d-1}. \tag{95}$$

D.5. The Analytical Solution to The Maxwell-Ampere Equation

Consider the 3-dimensional space $\mathbb{R}^3$, and the following current along the z-axis:

$$J(x) = I \cdot \delta^2(x_1 = 0, x_2 = 0, x_3 \in [z_1, z_2]). \tag{96}$$

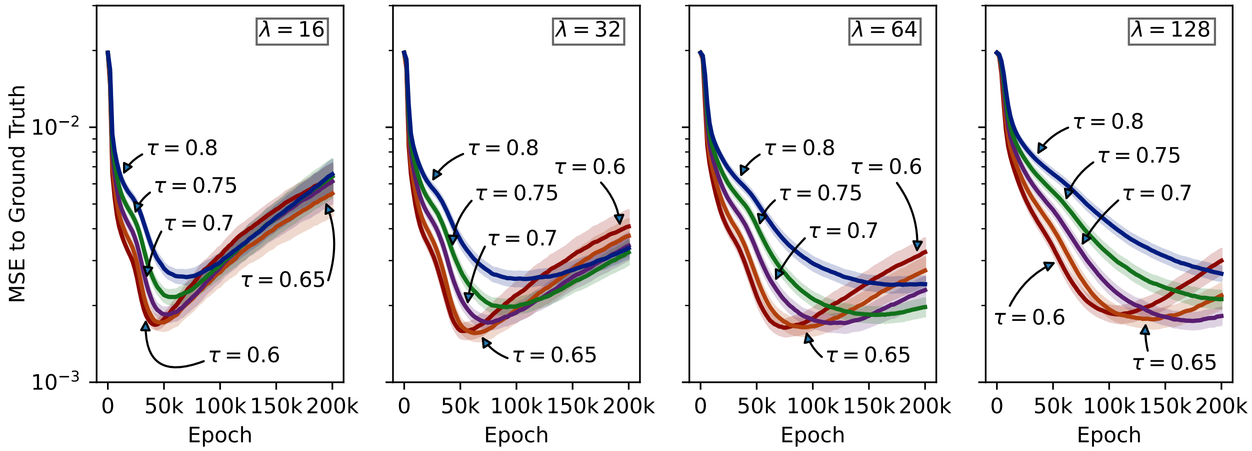

Figure 19. Studying the joint effect of target smoothing (τ) and regularization (λ) on the performance of the delayed target method in the Maxwell problem of Figure 5 in the main paper. Each plot corresponds to a fixed target regularization; we set λ to 16, 32, 64, and 128 in increasing order from the left to the right plots. Both hyper-parameters control the training speed. However, increasing the λ target regularization has a stabilizing effect on the training curves, whereas tuning the target smoothing τ mainly improves the peak performance.

The analytical solution to the $\nabla \times A = B$ and $\nabla \times B = J$ system described in Example 2.2 can be expressed as

$$A = \frac{-I}{4\pi} \cdot \log\left(\frac{(z_2 - x_3) + \sqrt{x_1^2 + x_2^2 + (z_2 - x_3)^2}}{(z_1 - x_3) + \sqrt{x_1^2 + x_2^2 + (z_1 - x_3)^2}}\right) \Big[0, 0, 1\Big]^{\mathrm{T}}, \tag{97}$$

and

$$B = \frac{-I}{4\pi \cdot \sqrt{x_1^2 + x_2^2}} \cdot \left(\frac{z_2 - x_3}{\sqrt{x_1^2 + x_2^2 + (z_2 - x_3)^2}} - \frac{z_1 - x_3}{\sqrt{x_1^2 + x_2^2 + (z_1 - x_3)^2}}\right) \cdot \begin{bmatrix} \frac{-x_2}{\sqrt{x_1^2+x_2^2}} \\ \frac{x_1}{\sqrt{x_1^2+x_2^2}} \\ 0 \end{bmatrix}. \tag{98}$$

D.6. Random Effect Matching

Random effects (random number generators seed; batch ordering; parameter initialization; and so on) complicate the study by creating variance in the measured statistics. We use a matching procedure (so that the baseline and the proposed models share the same values of all random effects) to control this variance. As long as one does not search for random effects that yield a desired outcome (we did not), this yields an unbiased estimate of the improvement. Each experiment is repeated 100 times to obtain confidence intervals. Note that (1) confidence intervals are small, and (2) experiments over many settings yield consistent results.

D.7. Training Hyper-parameters

We employed 3-5 layer perceptrons as our deep neural network, using 64 hidden neural units in each layer, and either the SiLU or $\tanh$ activation functions. We trained our networks using the Adam (Kingma & Ba, 2014) variant of the stochastic gradient descent algorithm under a learning rate of 0.001. We afforded each method and configuration 1000 function evaluations for each epoch. Table 4 provides a summary of these hyper-parameters along with the volume and surface point. In all figures where the ground truth MSE was plotted against a hyper-parameter, the epoch with the minimal MSE value for each method was picked to summarize the training curve (rather than the last epoch). All heatmaps show the average prediction of each method across different trainings and randomization seeds, except in Figure 2 of the main paper, where we hand-picked a single representative training for the heatmap visualizations to better illustrate the behaviors of the deterministic and double-sampling models.

In the high-dimensional Poisson problems of Figure 4 in the main paper, we used a 5-layer MLP with the SiLU activation for all methods. For the delayed target method, we set $\tau = 0.996$ and $\lambda = 4$, with $M = 500$, 250, and 100 for the $N = 1$,

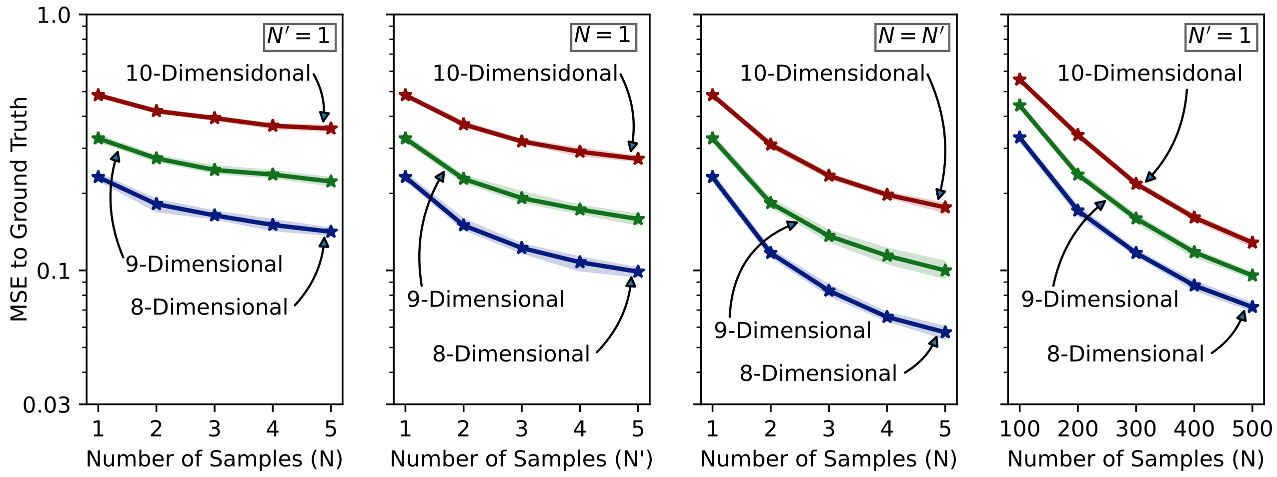

Figure 20. The effect of increasing the number of target samples on the solution quality in 8-, 9-, and 10-dimensional Poisson problems of Figure 4 of the main paper. The three left subplots show various running configurations of the delayed target method with increased sample sizes. The leftmost plot shows the effect of increasing N while keeping $N' = 1$, whereas the second plot from the left shows the effect of increasing N' while keeping $N = 1$. The third plot from the left shows the effect of increasing both N and N' equally. Increasing N' (i.e., the number of main model samples parameterized by θ) seems to be more effective than increasing N (i.e., the number of target model samples parameterized by θ^{Target}). Also, notice that all of the (1) $N = 5$ and $N' = 1$, (2) $N = 1$ and $N' = 5$, and (3) $N = N' = 3$ configurations have the same $N + N' = 6$ total value, yet, the last configuration yields the best performance. The rightmost plot shows the standard training method with 100 to 500 target samples.

10, and 100 curves, respectively. We used second-order quadrature with Smolyak sparse grids which determined N for each problem dimension; Gaussian quadrature defined N between 2 and 189, and Leja quadrature defined N between 2 and 54 for various dimensions. For the QMC method, we used the additive recursion rule for generating quasi-random sequences. In the Maxwell problem, we used a 5-layer MLP with the $\tanh$ activation for all methods, and we set $\tau = 0.75$, $\lambda = 64$, and $M = 1000$ for the delayed target method. In the Smoluchowski problems, we used a 3-layer MLP with the SiLU activation for all methods, and we set $\tau = 0.99$ and $\lambda = 1$ for the delayed target method. The detailed hyper-parameter configurations for each problem can be found in the paper's code base.

D.8. Evaluation Profiles

In high-dimensional problems, we found (1) the choice of the evaluation distribution, (2) the output pre-processing, and (3) the specific deterministic or stochastic performance estimator to be important for the results to be meaningful across different methods and problem dimensions.

Many evaluation distributions may seem reasonable for our unit-charge Poisson problems in Figure 4 of the main paper. For instance, sampling points uniformly from (1) the $[-1, 1]^d$ cube, (2) the unit ball centered at zero, (3) the normal distribution, and (4) the training volumes (i.e., the randomly sampled balls used to enforce the divergence theorem) all seem like reasonable choices. While all choices may yield similar results on low-dimensional problems, we found this choice to be more influential in higher-dimensional problems. In particular, sampling points uniformly from the training volumes yields the most meaningful proxy for comparing method performances as it follows the training distribution closely.

The second factor is the output pre-processing before computing the mean-squared error. Having a boundary condition may theoretically guarantee a unique solution to the PDE of interest. That being said, strong enforcement of a boundary condition can have a confounding effect on the evaluation dynamics; note that the boundary condition loss term can interact with the main loss and the induced excess variance in Equation (25), which may emphasize or hide a method's vulnerability to the excess variances problem. Furthermore, the ground truth solutions' scale can vary between dimensions. For this reason, we avoided enforcing the boundary condition too strongly to allow each method to demonstrate its unconstrained behavior. Moreover, we normalized the model and the ground truth outputs before computing the mean-squared error between them, such that they both have a zero empirical mean and unit empirical variance. We noticed that the training curves for all methods in high-dimensional problems tend to be overly noisy in the absence of this normalization pre-processing.

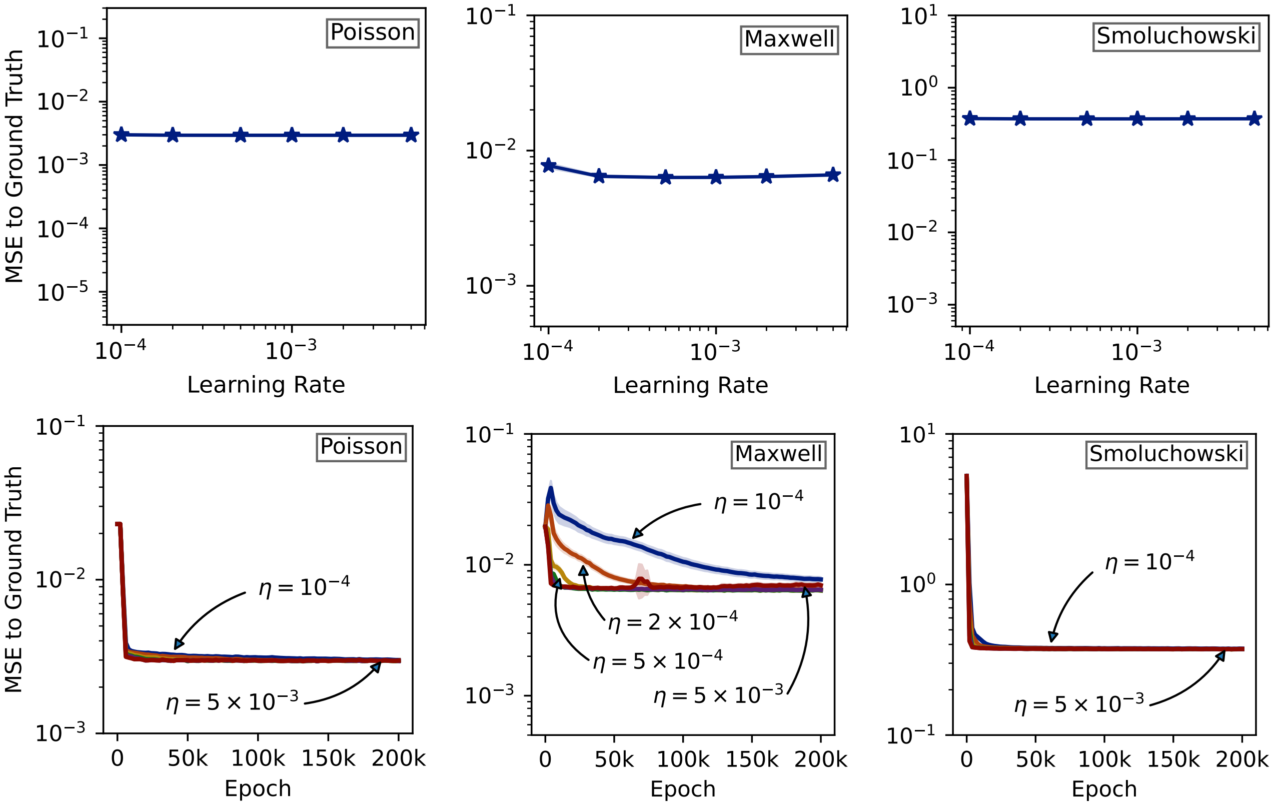

Figure 21. Ablating the effect of optimization learning rate on the standard training method with $N = 1$. *The top panel* shows the MSE to ground truth with respect to the learning rate. *The bottom panel* shows the corresponding training curves to the top panel. *The left, middle, and right columns* correspond to the 2D Poisson, 3D Maxwell-Ampere, and 2D Smoluchowski problems of Figures 1, 5, and 6 of the main paper, respectively.

Practically, this normalization step can be implemented by having the models "calibrated" through proper output shifting and scaling before evaluation. Since these calibration statistics are only two scalars for each model, this calibration assumption is not an overly unrealistic setup.

Finally, a stochastic i.i.d. sampling procedure of the evaluation points may not yield the most accurate results. Our training volume distribution is rotation invariant; in a spherical coordinate system, the joint distribution of points can be factored into two independent radii and angles distributions:

$$P(x) = P_r(r_x) \cdot P_\phi(\phi_x), \tag{104}$$

where $r_x = \|x\|_2$ and $\phi_x = \frac{x}{\|x\|_2}$ can express the x evaluation point in a spherical coordinate system. In particular, we found that sampling the point radii and angles independently and forming a grid can yield a robust estimator. Algorithm 2 details this procedure. Notice that this process is only applicable to rotation-invariant distributions, that is, distributions where the radii and angles are independent random variables as described in Equation (104). Our training volumes satisfy this condition. Algorithm 2 is statistically consistent, meaning that with $s, q, t \to \infty$, the estimated $\mathcal{L}_e$ is guaranteed to be accurate in the limit. While choosing a finite q may result in a small bias, it allows our performance estimator to be highly robust to outliers.

We set both of the q and s grid dimensions to be 500 in Algorithm 2. This results in an evaluation sample size of 2.5×10^5 points for each model. The auxiliary sample size t was set to a large value of 10^4. To reduce the randomization effects, we matched the random effects in Algorithm 2 for all methods; in other words, we used the same $\tilde{r}_i$ and $\tilde{z}_j$ for all evaluations. This allowed us to obtain a robust performance estimate across all methods, training iterations, and problem dimensions.

For the Maxwell-Ampere problem, we sampled the evaluation points in an i.i.d. manner from the training volumes. As for pre-processing, we only subtracted their empirical means from the model and ground truth solutions before computing the

Algorithm 2 The Robust Performance Estimator for the High-Dimensional Poisson Problem

```
Require: The number of radii quantiles $q$ and the number of spherical angles $s$.
    The resulting evaluation sample size will be $e := qs$.
Require: The space dimensionality $d$ and the auxilary sample size $t$.
Sample $t$ points from the training volumes in an i.i.d. manner, and denote their radii as $\{r_1, r_2, \cdots, r_t\}$.
Find the $(\frac{1}{2q}, \frac{3}{2q}, \cdots, \frac{2q-1}{2q})$ quantiles of the $\{r_1, r_2, \cdots, r_t\}$ population and name them $\{\tilde{r}_1, \cdots, \tilde{r}_q\}$.
Sample $z_1, z_2, \cdots, z_s$ from a $d$-dimensional normal distribution in an i.i.d. mannaer.
Define the evaluation angles:
    $$\tilde{z}_j := \frac{z_j}{\|z_j\|_2} \qquad \forall 1 \leq j \leq s. \tag{99}$$
Define the evaluation points grid:
    $$E := \{\tilde{r}_i \cdot \tilde{z}_j | 1 \leq i \leq q,\ 1 \leq j \leq s\} = \{x_1, x_2, \cdots, x_e\}. \tag{100}$$
    Note the size of the evaluation set $e$ is the product of the $q$ and $s$ sample sizes.
Evaluate the model and the ground truth solution on the evaluation set:
    $$\forall 1 \leq k \leq e : \qquad a_k := f_\theta(x_k), \qquad b_k := f^*(x_k). \tag{101}$$
Normalize the model and ground truth solutions:
    $$\forall 1 \leq k \leq e : \qquad \tilde{a}_k := \frac{a_k - \mathbb{E}_{i\sim\text{Unif}[1,e]}[a_i]}{\sqrt{\mathbb{V}_{i\sim\text{Unif}[1,e]}[a_i]}}, \qquad \tilde{b}_k := \frac{b_k - \mathbb{E}_{i\sim\text{Unif}[1,e]}[b_i]}{\sqrt{\mathbb{V}_{i\sim\text{Unif}[1,e]}[b_i]}}. \tag{102}$$
Return the MSE between the $\tilde{a}_k$ and $\tilde{b}_k$ values:
    $$\mathcal{L}_e := \frac{1}{e}\sum_{k=1}^{e}(\tilde{a}_k - \tilde{b}_k)^2. \tag{103}$$
```

mean squared error between them (i.e., no scaling was performed). No pre-processing was performed for the Smoluchowski problems as solving this PDE requires the initial conditions to be strongly enforced.

E. Computational Requirements

Assuming θ is d-dimensional and we use a deep feed-forward perceptron network, the required resources to run the delayed target method are detailed in Table 5. As a result, the delayed target method requires the following total computational and dynamic memory requirements per iteration:

$$C_{N,d}^{\text{DT}} = (c_5 + c_9)Nd + c_4N + (c_3 + c_9' + c_{10})d + c_6 + c_7 + c_8, \tag{105}$$

$$M_{N,d}^{\text{DT}} = (d_5 + d_9)Nd + d_4N + d_3d + d_6 + d_7 + d_8 + d_{10}. \tag{106}$$

In comparison, the standard method skips steps 7, 8, and 10, and therefore requires

$$C_{N,d} = (c_5 + c_9)Nd + c_4N + (c_3 + c_9')d + c_6, \tag{107}$$

$$M_{N,d} = (d_5 + d_9)Nd + d_4N + d_3d + d_6. \tag{108}$$

This means that both the standard and delayed target methods have per-step computational cost and dynamic memory usage that is dominated by the same $O(Nd)$ terms. Overall we expect the costs to be similar, with up to a doubling of per-step cost for the delayed target method because of the need for two networks (the main and target networks).

Hyper-Parameter	Value
Randomization Seeds	100
Learning Rate	0.001
Optimizer	Adam
Epoch Function Evaluations	1000
Training Epochs	200000
Network Depth	3-5 Layers
Network Width	64
Network Activation	SiLU or $\tanh$

Hyper-Parameter	Value
Problem Dimensions	1, 2, and 3
Initial Condition Weight	1
Initial Condition Time	0
Initial Condition Points	Unif$([0, 1])$
Time Distribution	Unif$([0, 1])$
Particle Size Distribution	Unif$([0, 1])$
Ground Truth Integrator	Euler
Ground Truth Grid Size	10000

Hyper-Parameter	Value
Problem Dimension	2
Number of Poisson Charges	3
Integration Volumes	Balls
Volume Center Distribution	Unif$([-1, 1])$
Volume Radius Distribution	Unif$([0.1, 1.5])$

Hyper-Parameter	Value
Problem Dimension	3
Wire Segments	4
Integration Volumes	2D Disks
Volume Center Distribution	Unit Ball
Volume Area Distribution	Unif$([0, 1])$

Table 4. A summary of the problem-specific hyper-parameters. *The top left table* represents the common settings used in all experiments. *The top right, bottom left, and bottom right* tables correspond to the Smoluchowski, Poisson, and Maxwell problems, respectively. In the high-dimensional Poisson problems, we defined a single charge at the origin, and the integration volumes were balls where (1) their centers were uniformly distributed inside the unit ball, and (2) their volumes followed a uniform distribution between zero and the volume of a unit ball.

F. Related work

F.1. Scientific Applications of Integro-Differential PDEs

Integro-differential PDEs arise in many areas such as quantum physics (Laskin, 2000; 2002; Elgart & Schlein, 2007; Lieb & Yau, 1987), visco-elastic fluid dynamics (Constantin, 2005; Caffarelli et al., 2011; Caffarelli & Vasseur, 2010a;b), nuclear reaction physics (Bern et al., 1994), mathematical finance (Nolan, 1999; Ros Oton, 2014), ecology (Humphries et al., 2010; Cabré & Roquejoffre, 2013; Reynolds & Rhodes, 2009; Viswanathan et al., 1996), elasticity and material modeling (Toland, 1997; Lu, 2005), particle system evolutions (Chapman et al., 1996; Weinan, 1994; Giacomin & Lebowitz, 1997; Carrillo et al., 2011), aerosol modeling (Wang et al., 2022a), computed tomography (Wei et al., 2019), radiation transfer and wave propagation (Modest & Mazumder, 2021), grazing systems and epidemealogy (Lakshmikantham, 1995), and in the formulation of weak solutions with methods such as variational PINNs (Kharazmi et al., 2021). Weak solutions using the divergence or the curl theorems, or the Smoluchowski coagulation equation (Wang et al., 2022a) are a few representative forms we consider as examples for learning from integral losses.

F.2. Physics-Informed Networks

The original Physics-Informed Neural Network (PINN) was introduced in Raissi et al. (2019). Later, variational PINNs were introduced in Kharazmi et al. (2019). Variational PINNs introduced the notion of weak solutions using test functions into the original PINNs. This was later followed by hp-VPINNs (Kharazmi et al., 2021). In fact, integral forms appear in both VPINNs and hp-VPINNs, and delayed target methods could be used synergistically with these variational models to improve them. The double-sampling trick was originally introduced in reinforcement learning literature (Baird, 1995). In the context of PINNs, Guo et al. (2022) used this technique to address the Monte-Carlo loss estimation problem in fractional PDEs.

Conservative PINNs (or cPINNs for short) were also proposed and used to solve physical systems with conservation

Line	Computational Cost	Memory Requirement
4	$c_3 \cdot d$	$d_3 \cdot d$
5	$c_4 \cdot N$	$d_4 \cdot N$
6	$c_5 \cdot N \cdot d$	$d_5 \cdot N \cdot d$
7	c_6	d_6
8	c_7	d_7
9	c_8	d_8
10	$c_9 \cdot N \cdot d + c'_9 K$	$d_9 \cdot N \cdot d$
11	$c_{10} d$	d_{10}

Table 5. The computational requirements of running Algorithm 1 of the main paper. The left column denotes the corresponding line in the algorithmic description. The middle and the right columns describe the computational cost and dynamic memory requirements, respectively.

laws (Jagtap et al., 2020) and Mao et al. (2020) examined the application of PINNs to high-speed flows. Many other papers attempted to scale and solve the fundamental problems with PINNs, for example, using domain decomposition techniques (Shukla et al., 2021; Li et al., 2019), the causality views (Wang et al., 2022b), and neural operators (Li et al., 2020). Reducing the bias of the estimated training loss is a general topic in machine and reinforcement learning (Sutton, 1988; Ghaffari et al., 2022; Arazo et al., 2020).

F.3. Bootstrapping Neural Networks

In general, the delayed target strategies and bootstrapping neural models, such as the TD-learning method, have been looked at in multiple contexts such as reinforcement or semi-supervised learning. The TD-learning method is an early example of this family (Sutton, 1984) and it has been analyzed extensively in prior work (Dayan, 1992; Tsitsiklis & Van Roy, 1997; Baird, 1995; Li, 2008; Schoknecht & Merke, 2003). Time and time again, TD-learning has proven preferable over the ordinary MSE loss minimization (known as the Bellman residual minimization) (Saleh & Jiang, 2019; Fujimoto et al., 2022; Yin et al., 2022; Chen et al., 2021). The deep Q-networks proposed a practical adaptation of this methodology (Mnih et al., 2015), which has been complemented in the TD3 method (Fujimoto et al., 2018).

Another example is the recent trend of semi-supervised learning, where teacher-student frameworks result in accuracy improvements of classification models by pseudo-labelling unlabelled examples for training (Hinton et al., 2015; Pham et al., 2021; Arazo et al., 2020; Lee et al., 2013). While a small number of recent theoretical insights exist on why semi-supervised learning does not produce trivially incorrect solutions (Tian et al., 2021), a wealth of theoretical literature analyzed the ability and shortcomings of TD-learning methods to solve such problems.

G. Recommendations and Limitations

From the numerical examples, we consistently see that the delayed taråget method shows superior performance over the other methods. However, this also has limitations, as this method is more temperamental than the standard trainings, and may require careful specification of hyper-parameters such as λ and τ, as we showed in Figure 3 of the main paper. Furthermore, we used large mini-batch sizes in conjunction with small N values. While stochastic gradient descent can often be more effective with small mini-batch sizes, adaptively tuning the target smoothing and regularization weights to always stabilize the main and target parameters in the delayed target method under small mini-batch sizes and highly stochastic problems is another direction for future work.

Our work solves the problem of learning from integral losses in physics-informed networks. We mostly considered singular and high-variance problems for benchmarking our methods. However, problems with integral losses can have broader applications in solving systems with incomplete observations and limited dataset sizes. This was beyond the scope of our work. Such applications may extend beyond the area of scientific learning and cover diverse applications within machine learning. We rigorously studied the utility of three methods for solving such systems. However, more algorithmic advances may be necessary to make the proposed methods robust and adaptive to the choice of algorithmic and problem-defining hyper-parameters. The delayed target method was shown to be capable of solving challenging problems through its approximate dynamic programming nature. However, we did not provide a systematic approach for identifying bottlenecks in case of failed trainings. Understanding the pathology of the studied methods is certainly a worthwhile future endeavor.