Number Theoretic Accelerated Learning of Physics-Informed Neural Networks

Takashi Matsubara[1], **Takaharu Yaguchi**[2]

[1]Hokkaido University
[2]Kobe University
matsubara@ist.hokudai.ac.jp, yaguchi@pearl.kobe-u.ac.jp

Abstract

Physics-informed neural networks solve partial differential equations by training neural networks. Since this method approximates infinite-dimensional PDE solutions with finite collocation points, minimizing discretization errors by selecting suitable points is essential for accelerating the learning process. Inspired by number theoretic methods for numerical analysis, we introduce good lattice training and periodization tricks, which ensure the conditions required by the theory. Our experiments demonstrate that GLT requires 2–7 times fewer collocation points, resulting in lower computational cost, while achieving competitive performance compared to typical sampling methods.

Introduction

Many real-world phenomena can be modeled as partial differential equations (PDEs), and solving PDEs has been a central topic in computational science. The applications include, but are not limited to, weather forecasting, vehicle design (Hirsch 2006), economic analysis (Achdou et al. 2014), and computer vision (Logan 2015). A PDE is expressed as $\mathcal{N}[u] = 0$, where $\mathcal{N}$ is a (possibly nonlinear) differential operator, and $u : \Omega \to \mathbb{R}$ is an unknown function on the domain $\Omega \subset \mathbb{R}^s$. For most PDEs that appear in physical simulations, the well-posedness (the uniqueness of the solution u and the continuous dependence on the initial and boundary conditions) has been well-studied and is typically guaranteed under certain conditions. To solve PDEs, various computational techniques have been explored, including finite difference methods, finite volume methods, and spectral methods (Furihata and Matsuo 2010; Morton and Mayers 2005; Thomas 1995). However, the development of computer architecture has become slower, leading to a growing need for computationally efficient alternatives. A promising approach is physics-informed neural networks (PINNs) (Raissi et al. 2019), which train a neural network by minimizing the physics-informed loss (Wang et al. 2022b; Wang and Perdikaris 2023). This is typically the squared error of the neural network's output $\tilde{u}$ from the PDE $\mathcal{N}[u] = 0$ averaged over a finite set of collocation points $\boldsymbol{x}_j$, $\frac{1}{N}\sum_{j=0}^{N-1} \|\mathcal{N}[\tilde{u}](\boldsymbol{x}_j)\|^2$, encouraging the output $\tilde{u}$ to satisfy the equation $\mathcal{N}[\tilde{u}](\boldsymbol{x}_j) = 0$.

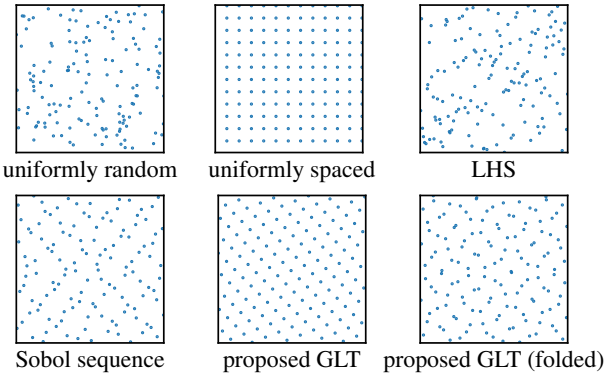

Figure 1: Examples of sampled collocation points. 128 points for the Sobol sequence, and 144 points for others.

However, the solutions u to PDEs are inherently infinite-dimensional, and any distance involving the output $\tilde{u}$ or the solution u needs to be defined by an integral over the domain Ω. In this regard, the physics-informed loss serves as a finite approximation to the squared 2-norm $|\mathcal{N}[\tilde{u}]|_2^2 = \int_{\boldsymbol{x}\in\Omega} \|\mathcal{N}[\tilde{u}](\boldsymbol{x})\|^2 \mathrm{d}\boldsymbol{x}$ on the function space $L^2(\Omega)$ for $\mathcal{N}[u] \in L^2(\Omega)$, and hence the discretization errors should affect the training efficiency. A smaller number N of collocation points leads to a less accurate approximation and inferior performance, while a larger number N increases the computational cost (Bihlo and Popovych 2022; Sharma and Shankar 2022). Despite the importance of selecting appropriate collocation points, insufficient emphasis has been placed on this aspect. Raissi et al. (2019), Zeng et al. (2023), and many other studies employed Latin hypercube sampling (LHS) to determine the collocation points. Alternative approaches include uniformly random sampling (i.e., the Monte Carlo method) (Jin et al. 2021; Krishnapriyan et al. 2022) and uniformly spaced sampling (Wang et al. 2021, 2022a). These methods are exemplified in Fig. 1.

In the field of numerical analysis, the relationship between integral approximation and collocation points has been extensively investigated. Accordingly, some studies have used quasi-Monte Carlo methods, specifically the Sobol sequence, which approximate integrals more accurately than

the Monte Carlo method (Lye et al. 2020; Longo et al. 2021; Mishra and Molinaro 2021). For further improvement, this paper proposes *good lattice training (GLT)* for PINNs and their variants, such as the competitive PINN (CPINN) (Zeng et al. 2023) and physics-informed neural operator (Li et al. 2021; Rosofsky et al. 2023). The GLT is inspired by number theoretic methods for numerical analysis, providing an optimal set of collocation points depending on the initial and boundary conditions, as shown in Fig. 1. Our experiments demonstrate that the proposed GLT requires far fewer collocation points than comparison methods while achieving similar errors, significantly reducing computational cost. The contribution and significance of the proposed GLT are threefold.

Computationally Efficient: The proposed GLT offers an optimal set of collocation points to compute a loss function that can be regarded as a finite approximation to an integral over the domain, such as the physics-informed loss, if the activation functions of the neural networks are smooth enough. It requires significantly fewer collocation points to achieve accuracy of solutions and system identifications comparable to other methods, or can achieve lower errors with the same computational budget.

Applicable to PINNs Variants: As the proposed GLT changes only the collocation points, it can be applied to various variants of PINNs without modifying the learning algorithm or objective function. In this study, we investigate a specific variant, the CPINNs (Zeng et al. 2023), and demonstrate that CPINNs using the proposed GLT achieve superior convergence speed with significantly fewer collocation points than CPINNs using LHS.

Theoretically Solid: Number theory provides a theoretical basis for the efficacy of the proposed GLT. Existing methods based on quasi-Monte Carlo methods are inferior to the proposed GLT in theoretical performance, or at least require the prior knowledge about the smoothness α of the solution u and careful adjustments of hyperparameters (Longo et al. 2021). On the other hand, the proposed GLT is free from these prior knowledge or adjustments and achieves better performances depending on the smoothness α and the neural network, which is a significant advantage.

Related Work

Neural networks are a powerful tool for processing information and have achieved significant success in various fields (He et al. 2016; Vaswani et al. 2017), including black-box system identification, that is, to learn the dynamics of physical phenomena from data and predict their future behaviors (Chen et al. 2018; Chen, Billings, and Grant 1990; Wang and Lin 1998). By integrating knowledge from analytical mechanics, neural networks can learn dynamics that adheres to physical laws and even uncover these laws from data (Finzi et al. 2020; Greydanus, Dzamba, and Yosinski 2019; Matsubara and Yaguchi 2023).

Neural networks have also gained attention as computational tools for solving differential equations, particularly PDEs (Dissanayake and Phan-Thien 1994; Lagaris, Likas, and Fotiadis 1998). Recently, Raissi et al. (2019) introduced an elegant refinement to this approach and named it PINNs. The key concept behind PINNs is the physics-informed loss (Wang et al. 2022b; Wang and Perdikaris 2023). This loss function evaluates the extent to which the output $\tilde{u}$ of the neural network satisfies a given PDE $\mathcal{N}[u] = 0$ and its associated initial and boundary conditions $\mathcal{B}[u] = 0$. The physics-informed loss can be integrated into other models like DeepONet (Lu et al. 2019; Wang and Perdikaris 2023) or used for white-box system identifications (that is, adjusting the parameters of known PDEs so that their solutions fit observations).

PINNs are applied to various PDEs (Bihlo and Popovych 2022; Jin et al. 2021; Mao et al. 2020), with significant efforts in improving learning algorithms and objective functions (Hao et al. 2023; Heldmann et al. 2023; Lu et al. 2022; Pokkunuru et al. 2023; Sharma and Shankar 2022; Zeng et al. 2023). Objective functions are generally based on PDEs evaluated at a finite set of collocation points rather than data. Bihlo and Popovych (2022); Sharma and Shankar (2022) have shown a trade-off between the number of collocation points (and hence computational cost) and the accuracy of the solution. Thus, selecting collocation points that efficiently cover the entire domain Ω is essential for achieving better results. Some studies have employed quasi-Monte Carlo methods, specifically the Sobol sequence, to determine the collocation points (Lye et al. 2020; Mishra and Molinaro 2021), but their effectiveness depends on knowledge of the solution's smoothness α and hyperparameter adjustments (Longo et al. 2021).

Method

Theoretical Error Estimate of PINNs For simplicity, we consider PDEs defined on an s-dimensional unit cube $[0, 1]^s$. PINNs employ a PDE that describes the target physical phenomena as loss functions. Specifically, first, an appropriate set of collocation points $L^* = \{\boldsymbol{x}_j \mid j = 0, \dots, N-1\}$ is determined, and then the sum of the residuals of the PDE at these points

$$\frac{1}{N}\sum_{j=0}^{N-1} \mathcal{P}[\tilde{u}](\boldsymbol{x}_j) = \frac{1}{N}\sum_{\boldsymbol{x}_j \in L^*} \mathcal{P}[\tilde{u}](\boldsymbol{x}_j), \qquad (1)$$

is minimized as a loss function, where $\mathcal{P}$ is a differential operator. The physics-informed loss satisfies $\mathcal{P}[\tilde{u}](\boldsymbol{x}) = \|\mathcal{N}[\tilde{u}](\boldsymbol{x})\|^2$. Then, the neural network's output $\tilde{u}$ becomes an approximate solution of the PDE. However, for $\tilde{u}$ to be the exact solution, the loss function should be 0 for all $\boldsymbol{x} \in \Omega$, not just at collocation points. Therefore, the following integral must be minimized as the loss function:

$$\int_{[0,1]^s} \mathcal{P}[\tilde{u}](\boldsymbol{x}) \mathrm{d}\boldsymbol{x}. \qquad (2)$$

In other words, the practical minimization of (1) essentially minimizes the approximation of (2) with the expectation that (2) will be small enough, and hence $\tilde{u}$ becomes an accurate approximation to the exact solution.

More precisely, we show the following theorem, which is an improvement of an existing error analysis (Mishra and Molinaro 2023) in the sense that the approximation error bound of neural networks is considered.

Theorem 1. *Suppose that the class of neural networks used for PINNs includes an ε_1-approximator $\tilde{u}_{\rm opt}$ to the exact solution u^* to the PDE $\mathcal{N}[u] = 0$: $\|u^* - \tilde{u}_{\rm opt}\| \le \varepsilon_1$, and that (1) is an ε_2-approximation of (2) for the approximated solution $\tilde{u}$ and also for $\tilde{u}_{\rm opt}$: $|\int_{[0,1]^s} \mathcal{P}[u](\boldsymbol{x})\mathrm{d}\boldsymbol{x} - \frac{1}{N}\sum_{\boldsymbol{x}_j \in L^*} \mathcal{P}[u](\boldsymbol{x}_j)| \le \varepsilon_2$ for $u = \tilde{u}$ and $u = \tilde{u}_{\rm opt}$. Suppose also that there exist $c_{\rm p} > 0$ and $c_{\rm L} > 0$ such that $\frac{1}{c_{\rm p}}\|u - v\| \le \|\mathcal{N}[u] - \mathcal{N}[v]\| \le c_{\rm L}\|u - v\|$. Then,*

$$\|u^* - \tilde{u}\| \le (1 + c_{\rm p}c_{\rm L})\varepsilon_1 + c_{\rm p}\sqrt{\tfrac{1}{N}\textstyle\sum_{\boldsymbol{x}_j \in L^*} \mathcal{P}[\tilde{u}](\boldsymbol{x}_j) + \varepsilon_2}.$$

For a proof, see Appendix "Theoretical Background." ε_1 is determined by the architecture of the network and the function space to which the solution belongs. For example, approximation rates of neural networks in Sobolev spaces are given in Gühring and Raslan (2021). This explains why increasing the number of collocation points beyond a certain point does not further reduce the error. ε_2 depends on the accuracy of the approximation of the integral. In this paper, we investigate a training method that easily gives small ε_2.

One standard strategy often used in practice is to set $\boldsymbol{x}_j$'s to be uniformly distributed random numbers, which can be interpreted as the Monte Carlo approximation of the integral (2). As is widely known, the Monte Carlo method can approximate the integral within an error of $O(1/N^{\frac{1}{2}})$ independently from the number s of dimensions (Sloan and Joe 1994). However, most PDEs for physical simulations are two to four-dimensional, incorporating a three-dimensional space and a one-dimensional time. Hence, in this paper, we propose a sampling strategy specialized for low-dimensional cases, inspired by number-theoretic numerical analysis.

Note that some variants, such as CPINN (Zeng et al. 2023), have proposed alternative objective functions. We hereafter denote any variant of physics-informed loss by (1), without loss of generality, as long as it can be regarded as a finite approximation to an integral over a domain, (2).

Good Lattice Training In this section, we propose the *good lattice training (GLT)*, in which a number theoretic numerical analysis is used to accelerate the training of PINNs (Niederreiter 1992; Sloan and Joe 1994; Zaremba 1972). In the following, we use some tools from this theory.

While our target is a PDE on the unit cube $[0, 1]^s$, we now treat the loss function $\mathcal{P}[\tilde{u}]$ as periodic on $\mathbb{R}^s$ with a period of 1. Then, we define a lattice.

Definition 2 (Sloan and Joe (1994))**.** A lattice L in $\mathbb{R}^s$ is defined as a finite set of points in $\mathbb{R}^s$ that is closed under addition and subtraction.

Given a lattice L, the set of collocation points is defined as $L^* = \{\boldsymbol{x}_j \mid j = 0, \dots, N-1\} \coloneqq \{\text{the decimal part of } \boldsymbol{x} \mid \boldsymbol{x} \in L\} \in [0, 1]^s$. Considering that the loss function to be minimized is (2), it is desirable to determine the lattice L (and hence the set of collocation points $\boldsymbol{x}_j$'s) so that the difference $|(2) - (1)|$ of the two functions is minimized.

Suppose that $\varepsilon(\boldsymbol{x}) \coloneqq \mathcal{P}[\tilde{u}](\boldsymbol{x})$ is smooth enough, admitting the Fourier series expansion:

$$\varepsilon(\boldsymbol{x}) \coloneqq \mathcal{P}[\tilde{u}](\boldsymbol{x}) = \textstyle\sum_{\boldsymbol{h}} \hat{\varepsilon}(\boldsymbol{h}) \exp(2\pi \mathrm{i} \boldsymbol{h} \cdot \boldsymbol{x}),$$

where i denotes the imaginary unit and $\boldsymbol{h} = (h_1, h_2, \dots, h_s) \in \mathbb{Z}^s$. Substituting this into (1) yields

$$|(2) - (1)| = \left| \tfrac{1}{N} \textstyle\sum_{j=0}^{N-1} \sum_{\boldsymbol{h} \in \mathbb{Z}^s, \boldsymbol{h} \neq 0} \hat{\varepsilon}(\boldsymbol{h}) \exp(2\pi \mathrm{i} \boldsymbol{h} \cdot \boldsymbol{x}_j) \right|, \tag{3}$$

because the Fourier mode of $\boldsymbol{h} = 0$ is equal to the integral $\int_{[0,1]^s} \varepsilon(\boldsymbol{x})\mathrm{d}\boldsymbol{x}$. Before optimizing (3), the dual lattice of lattice L and an insightful lemma are introduced as follows.

Definition 3 (Zaremba (1972))**.** A dual lattice $L^\top$ of a lattice L is defined as $L^\top \coloneqq \{\boldsymbol{h} \in \mathbb{R}^s \mid \boldsymbol{h} \cdot \boldsymbol{x} \in \mathbb{Z},\ \forall \boldsymbol{x} \in L\}$.

Lemma 4 (Zaremba (1972))**.** *For $\boldsymbol{h} \in \mathbb{Z}^s$, it holds that*

$$\tfrac{1}{N} \textstyle\sum_{j=0}^{N-1} \exp(2\pi \mathrm{i} \boldsymbol{h} \cdot \boldsymbol{x}_j) = \begin{cases} 1 & (\boldsymbol{h} \in L^\top) \\ 0 & (\text{otherwise.}) \end{cases}$$

Lemma 4 follows directly from the properties of Fourier series. Based on this lemma, we restrict the lattice point L to the form $\{\boldsymbol{x} \mid \boldsymbol{x} = \frac{j}{N}\boldsymbol{z} \text{ for } j \in \mathbb{Z}\}$ with a fixed integer vector $\boldsymbol{z}$; the set L^* of collocation points is $\{\text{the decimal part of } \frac{j}{N}\boldsymbol{z} \mid j = 0, \dots, N-1\}$. Then, instead of searching $\boldsymbol{x}_j$'s, a vector $\boldsymbol{z}$ is searched. By restricting to this form, $\boldsymbol{x}_j$'s can be obtained automatically from a given $\boldsymbol{z}$, and hence the optimal collocation points $\boldsymbol{x}_j$'s do not need to be stored as a table of numbers, making a significant advantage in implementation. Another advantage is theoretical; the optimization problem of the collocation points can be reformulated in a number theoretic way. In fact, for L as shown above, it is confirmed that $L^\top = \{\boldsymbol{h} \mid \boldsymbol{h} \cdot \boldsymbol{z} \equiv 0 \pmod N\}$. If $\boldsymbol{h} \cdot \boldsymbol{z} \equiv 0 \pmod N$ then there exists an $m \in \mathbb{Z}$ such that $\boldsymbol{h} \cdot \boldsymbol{z} = mN$ and hence $\frac{j}{N}\boldsymbol{h} \cdot \boldsymbol{z} = mj \in \mathbb{Z}$. Conversely, if $\boldsymbol{h} \cdot \boldsymbol{z} \not\equiv 0 \pmod N$, clearly $\frac{1}{N}\boldsymbol{h} \cdot \boldsymbol{z} \notin \mathbb{Z}$.

From the above lemma,

$$(3) \le \textstyle\sum_{\boldsymbol{h} \in \mathbb{Z}^s, \boldsymbol{h} \neq 0, \boldsymbol{h} \cdot \boldsymbol{z} \equiv 0 \pmod N} |\hat{\varepsilon}(\boldsymbol{h})|, \tag{4}$$

and hence the collocation points $\boldsymbol{x}_j$'s should be determined so that (4) becomes small. This problem is a number theoretic problem in the sense that it is a minimization problem of finding an integer vector $\boldsymbol{h}$ subject to the condition $\boldsymbol{h} \cdot \boldsymbol{z} \equiv 0 \pmod N$. This problem has been considered in the field of number theoretic numerical analysis. In particular, optimal solutions have been investigated for integrands in the Korobov spaces, which are spaces of functions that satisfy a certain smoothness condition.

Definition 5 (Zaremba (1972))**.** The function space that is defined as $E_\alpha = \{f : [0,1]^s \to \mathbb{R} \mid {}^\exists c, |\hat{f}(\boldsymbol{h})| \le \frac{c}{(\bar{h}_1 \bar{h}_2 \cdots \bar{h}_s)^\alpha}\}$ is called the Korobov space, where $\hat{f}(\boldsymbol{h})$ is the Fourier coefficients of f and $\bar{k} = \max(1, |k|)$ for $k \in \mathbb{R}$.

It is known that if α is an integer, for a function f to be in E_α, it is sufficient that f has continuous partial derivatives $\frac{\partial^{q_1+q_2+\cdots+q_s}}{\partial_1^{q_1} \cdot \partial_2^{q_2} \cdots \partial_s^{q_s}} f, 0 \le q_k \le \alpha\ (k = 1, \dots, s)$. For example, if a function $f(x, y) : \mathbb{R}^2 \to \mathbb{R}$ has continuous f_x, f_y, f_{xy}, then $f \in E_1$. Hence, if $\mathcal{P}[\tilde{u}]$ and the neural network belong to Koborov space,

$$(4) \le \textstyle\sum_{\boldsymbol{h} \in \mathbb{Z}^s, \boldsymbol{h} \neq 0, \boldsymbol{h} \cdot \boldsymbol{z} \equiv 0 \pmod N} \frac{c}{(\bar{h}_1 \bar{h}_2 \cdots \bar{h}_s)^\alpha}. \tag{5}$$

Here, we introduce a theorem in the field of number theoretic numerical analysis:

Theorem 6 (Sloan and Joe (1994))**.** *For integers $N \geq 2$ and $s \geq 2$, there exists a $\boldsymbol{z} \in \mathbb{Z}^s$ such that*

$$P_\alpha(\boldsymbol{z}, N) \leq \frac{(2\log N)^{\alpha s}}{N^\alpha} + O\left(\frac{(\log N)^{\alpha s - 1}}{N^\alpha}\right).$$

for $P_\alpha(\boldsymbol{z}, N) = \frac{1}{(\bar{h}_1 \bar{h}_2 \cdots \bar{h}_s)^\alpha}$.

The main result of this paper is the following.

Theorem 7. *Suppose that the activation function of $\tilde{u}$ and hence $\tilde{u}$ itself are sufficiently smooth so that there exists an $\alpha > 0$ such that $\mathcal{P}[\tilde{u}] \in E_\alpha$. Then, for given integers $N \geq 2$ and $s \geq 2$, there exists an integer vector $\boldsymbol{z} \in \mathbb{Z}^s$ such that $L^* = \{$the decimal part of $\frac{j}{N}\boldsymbol{z} \mid j = 0, \ldots, N-1\}$ is a "good lattice" in the sense that*

$$\left| \int_{[0,1]^s} \mathcal{P}[\tilde{u}](\boldsymbol{x}) \mathrm{d}\boldsymbol{x} - \frac{1}{N} \sum_{\boldsymbol{x}_j \in L^*} \mathcal{P}[\tilde{u}](\boldsymbol{x}_j) \right| = O\left(\frac{(\log N)^{\alpha s}}{N^\alpha}\right). \tag{6}$$

Intuitively, if $\mathcal{P}[\tilde{u}]$ satisfies certain conditions, we can find a set L^* of collocation points with which the objective function (1) approximates the integral (2) only within an error of $O(\frac{(\log N)^{\alpha s}}{N^\alpha})$. This rate is much better than that of the uniformly random sampling (i.e., the Monte Carlo method), which is of $O(1/N^{\frac{1}{2}})$ (Sloan and Joe 1994), if the activation function of $\tilde{u}$ and hence the neural network $\tilde{u}$ itself are sufficiently smooth so that $\mathcal{P}[\tilde{u}] \in E_\alpha$ for a large α. Hence, in this paper, we call the training method that minimizes (1) for a lattice L satisfying (6) the *good lattice training (GLT)*.

While any set of collocation points that satisfies the above condition will have the same convergence rate, a set constructed by the vector $\boldsymbol{z} \in \mathbb{Z}^s$ that minimizes (5) leads to better accuracy. When $s = 2$, it is known that a good lattice can be constructed by setting $N = F_k$ and $\boldsymbol{z} = (1, F_{k-1})$, where F_k denotes the k-th Fibonacci number (Niederreiter 1992; Sloan and Joe 1994). In general, an algorithm exists that can determine the optimal $\boldsymbol{z}$ with a computational cost of $O(N^2)$. See Appendix "Theoretical Background" for more details. Also, we can retrieve the optimal $\boldsymbol{z}$ from numerical tables found in references, such as Fang and Wang (1994); Keng and Yuan (1981).

Periodization and Randomization Tricks The integrand $\mathcal{P}[\tilde{u}]$ of the loss function (2) does not always belong to the Korobov space E_α with high smoothness α. To align the proposed GLT with theoretical expectations, we propose periodization tricks for ensuring periodicity and smoothness.

Given an initial condition at time $t = 0$, the periodicity is ensured by extending the lattice twice as much along the time coordinate and folding it. Specifically, instead of t, we use $\hat{t}$ as the time coordinate that satisfies $t = 2\hat{t}$ if $\hat{t} < 0.5$ and $t = 2(1 - \hat{t})$ otherwise (see the lower right panel of Fig. 1, where the time is put on the horizontal axis). Also, while not mandatory, we combine the initial condition $u_0(\boldsymbol{x}_{/t})$ and the neural network's output $\tilde{u}(t, \ldots)$ as $\exp(-t)u_0(\boldsymbol{x}_{/t}) + (1 - \exp(-t))\tilde{u}(t, \boldsymbol{x}_{/t})$, thereby ensuring the initial condition, where $\boldsymbol{x}_{/t}$ denotes the set of coordinates except for the time coordinate t. A similar idea was proposed in Lagaris, Likas, and Fotiadis (1998), which however does not ensure the initial condition strictly. If a periodic boundary condition is given to the k-th space coordinate, we bypass learning it and instead map the coordinate x_k to a unit circle in two-dimensional space. Specifically, we map x_k to $(x_k^{(1)}, x_k^{(2)}) = (\cos(2\pi x_k), \sin(2\pi x_k))$, assuring the loss function $\mathcal{P}[\tilde{u}]$ to take the same value at the both edges ($x_k = 0$ and $x_k = 1$) and be periodic. Given a Dirichlet boundary condition $u = 0$ at $\partial\Omega$ to the k-th axis, we multiply the neural network's output $\tilde{u}(\ldots, x_k, \ldots)$ by $x_k(1 - x_k)$, and treat the result as the approximated solution. This ensures the Dirichlet boundary condition is met, and the loss function $\mathcal{P}[\tilde{u}]$ takes zero at the boundary $\partial\Omega$, thereby ensuring the periodicity. If a more complicated Dirichlet boundary condition is given, one can fold the lattice along the space coordinate in the same manner as the time coordinate and ensure the periodicity of the loss function $\mathcal{P}[\tilde{u}]$.

These periodization tricks aim to satisfy the periodicity conditions necessary for GLT to exhibit the performance shown in Theorem 7. However, they are also available for other sampling methods and potentially improve the practical performance by liberating them from the effort of learning initial and boundary conditions.

Since the GLT is grounded on the Fourier series foundation, it allows for the periodic shifting of the lattice. Hence, we randomize the collocation points as

$$L^* = \{\text{the decimal part of } \tfrac{j}{N}\boldsymbol{z} + \boldsymbol{r} \mid j = 0, \ldots, N-1\},$$

where $\boldsymbol{r}$ follows the uniform distribution over the unit cube $[0, 1]^s$. Our preliminary experiments confirmed that, if using the stochastic gradient descent (SGD) algorithm, resampling the random numbers $\boldsymbol{r}$ at each training iteration prevents the neural network from overfitting and improves training efficiency. We call this approach the randomization trick.

Limitations Not only is this true for the proposed GLT, but most strategies to determine collocation points are not directly applicable to non-rectangular or non-flat domain Ω (Shankar et al. 2018). To achieve the best performance, the PDEs should be transformed to such a domain by an appropriate coordinate transformation. See Knupp and Steinberg (2020); Thompson et al. (1985) for examples.

Previous studies on numerical analysis addressed the periodicity and smoothness conditions on the integrand by variable transformations (Sloan and Joe 1994) (see also Appendix "Theoretical Background"). However, our preliminary experiments confirmed that it did not perform optimally in typical problem settings. Intuitively, these variable transformations reduce the weights of regions that are difficult to integrate, suppressing the discretization error. This implies that, when used for training, the regions with small weights remain unlearned. As a viable alternative, we introduced the periodization tricks to ensure periodicity.

The performance depends on the smoothness of the physics-informed loss, and hence on the smoothness of the neural network and the true solution. See Appendix "Theoretical Background" for details.

Experiments and Results

Physics-Informed Neural Networks We modified the code from the official repository[1] of Raissi et al. (2019), the original paper on PINNs. We obtained the datasets of the nonlinear Schrödinger (NLS) equation, Korteweg–De Vries (KdV) equation, and Allen-Cahn (AC) equation from the repository. The NLS equation governs wave functions in quantum mechanics, while the KdV equation models shallow water waves, and the AC equation characterizes phase separation in co-polymer melts. These datasets provide numerical solutions to initial value problems with periodic boundary conditions. Although they contain numerical errors, we treated them as the true solutions u. These equations are nonlinear versions of hyperbolic or parabolic PDEs. Additionally, as a nonlinear version of an elliptic PDE, we created a dataset for s-dimensional Poisson's equation, which produces analytically solvable solutions with 2^s modes with the Dirichlet boundary condition. We examined the cases where $s \in \{2, 4\}$. See Appendix "Experimental Settings" for further information.

Unless otherwise stated, we followed the repository's experimental settings for the NLS equation. The physics-informed loss was defined as $\mathcal{P}[\tilde{u}] = \frac{1}{N}\sum_{j=0}^{N-1} \|\mathcal{N}[\tilde{u}](\boldsymbol{x}_j)\|^2$ given N collocation points $\{\boldsymbol{x}_j\}_{j=0}^{N-1}$. This can be regarded as a finite approximation to the squared 2-norm $|\mathcal{N}[\tilde{u}]|_2^2 = \int_\Omega \|\mathcal{N}[\tilde{u}](\boldsymbol{x})\|^2 \mathrm{d}\boldsymbol{x}$. The state of the NLS equation is complex; we simply treated it as a 2D real vector for training and used its absolute value for evaluation and visualization. Following Raissi et al. (2019), we evaluated the performance using the relative error, which is the normalized squared error $\mathcal{L}(\tilde{u}, u; \boldsymbol{x}_j) = (\sum_{j=0}^{N_e-1} \|\tilde{u}(\boldsymbol{x}_j) - u(\boldsymbol{x}_j)\|^2)^{\frac{1}{2}} / (\sum_{j=0}^{N_e-1} \|u(\boldsymbol{x}_j)\|^2)^{\frac{1}{2}}$ at predefined N_e collocation points $\{\boldsymbol{x}_j\}_{j=0}^{N_e-1}$. This is also a finite approximation to $|\tilde{u} - u|_2 / |u|_2$.

We applied the above periodization tricks to each test and model for a fair comparison. For Poisson's equation with $s = 2$, which gives the exact solutions, we followed the original learning strategy using the L-BFGS-B method preceded by the Adam optimizer (Kingma and Ba 2015) for 50,000 iterations to ensure precise convergence. For other datasets, which contain the numerical solutions, we trained PINNs using the Adam optimizer with cosine decay of a single cycle to zero (Loshchilov and Hutter 2017) for 200,000 iterations and sampled a different set of collocation points at each iteration. See Appendix "Experimental Settings" for details.

We determined the collocation points using uniformly random sampling, uniformly spaced sampling, LHS, the Sobol sequence, and the proposed GLT. For the GLT, we took the number N of collocation points and the corresponding integer vector $\boldsymbol{z}$ from numerical tables in (Fang and Wang 1994; Keng and Yuan 1981). We used the same values for uniformly random sampling and LHS to maintain consistency. For uniformly spaced sampling, we selected numbers $N = m^s$ for $m \in \mathbb{N}$ that were closest to a number used in the GLT, creating a unit cube of m points on each side. We additionally applied the randomization trick. For the Sobol sequence, we used $N = 2^m$ for $m \in \mathbb{N}$ due to its theoretical background. We conducted five trials for each number N and each method. All experiments were conducted using Python v3.7.16 and tensorflow v1.15.5 (Abadi et al. 2016) on servers with Intel Xeon Platinum 8368.

[1] https://github.com/maziarraissi/PINNs (MIT license)

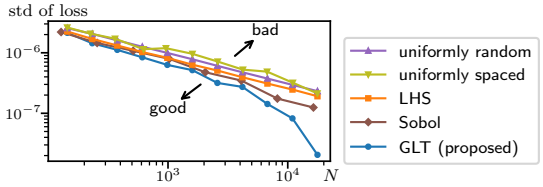

Figure 2: The number N of collocation points and the standard deviation of the physics-informed loss, which approximates the discretization error $|(2) - (1)|$.

Results: Accuracy of Physics-Informed Loss First, we evaluate the accuracy of the physics-informed loss (1) in approximating the integrated loss (2). Note that it is intractable to obtain the integrated loss (2), which motivates the present study. Instead, we use the standard deviation of the physics-informed loss (1) as an approximation to the discretization error $|(2) - (1)|$. This is because the average of the physics-informed loss (1) is assumed to converge to the integrated loss (2), and the standard deviation represents the average error. We trained the PINNs on the NLS equation with $N = 610$ collocation points determined by LHS. Then, we evaluated the physics-informed loss (1) for different methods and different numbers of collocation points. For each combination of method and number, we performed 10,000 trials and summarized their results in Fig. 2.

Except for the Sobol sequence and GLT, the other methods exhibit a similar trend, showing a linear reduction on the log-log plot. This trend aligns with the theoretical result that the convergence rate $O(1/N^{\frac{1}{2}})$ of the Monte Carlo method and that $O(1/N^{\frac{1}{s}})$ of the uniformly spaced sampling. The Sobol sequence shows a slightly faster reduction, and the GLT demonstrates a further accelerated reduction as the number N increases. This result implies that by using the GLT, the physics-informed loss (1) approximates the integrated loss (2) more accurately with the same number N of collocation points, leading to faster training and improved accuracy.

The Sobol sequence produces the discretization error of $O(\frac{(\log N)^s}{N})$ for smooth solutions u and neural networks, which is smaller than that $O(1/N^{\frac{1}{2}})$ of the Monte Carlo method for a large number N (Lye et al. 2020; Mishra and Molinaro 2021). As shown in Theorem 7, the proposed GLT produces the discretization error of $O(\frac{(\log N)^{\alpha s}}{N^\alpha})$, which is comparable to the Sobol sequence for solutions with $\alpha = 1$ and is much smaller for smoother solutions with $\alpha > 1$. These are several higher-order quasi-Monte Carlo methods, which potentially suppress the discretization errors for smooth solutions with $\alpha > 1$. However, these methods require the prior knowledge about the smoothness α and care-

	# of points $N^\dagger$					relative error $\mathcal{L}^\ddagger$				
	NLS	KdV	AC	Poisson		NLS	KdV	AC	Poisson	
				$s=2$	$s=4$				$s=2$	$s=4$
▲ uniformly random	>4,181	>4,181	4,181	>4,181	1,019	3.11	2.97	1.55	28.53	0.28
▼ uniformly spaced	2,601	4,225	>4,225	>4,225	>4,096	2.15	3.28	1.95	5.16	1437.12
■ LHS	>4,181	4,181	4,181	4,181	701	2.75	3.06	1.25	246.29	0.24
◆ Sobol	2,048	2,048	4,096	>4,096	1,024	2.05	2.52	1.22	14.74	1.22
● GLT (proposed)	**987**	**987**	**1,597**	**610**	**307**	**1.22**	**2.19**	**0.93**	**0.76**	**0.15**

† # of points N at competitive relative error $\mathcal{L}$ (under horizontal red line in Fig. 3).
‡ relative error $\mathcal{L}$ at competitive # of points N (on vertical green line in Fig. 3). Shown in the scale of 10^{-3}.

Table 1: Trade-Off between Number N of Collocation Points and Relative Error $\mathcal{L}$.

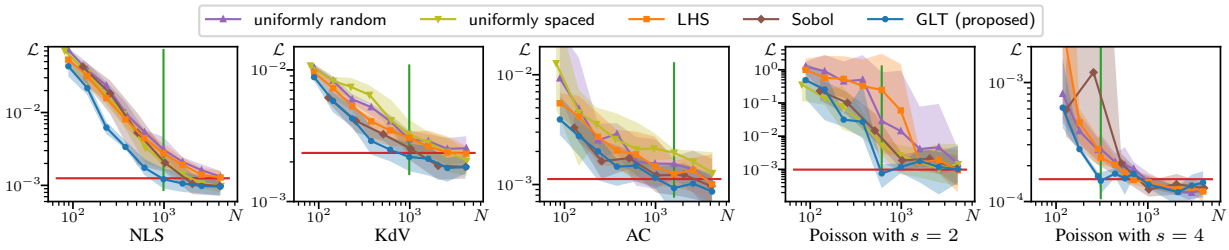

Figure 3: The results of PINNs. The number N of collocation points and the relative error $\mathcal{L}$.

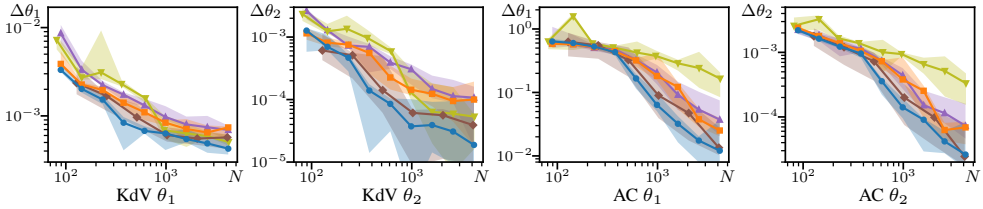

Figure 4: The results of system identification. The number N of collocation points and the relative error $\Delta\theta$ of the learnable parameter θ. The legend can be found in Fig. 3.

ful adjustments of hyperparameters (Longo et al. 2021). On the other hand, the proposed GLT is free from these prior knowledge or adjustments and achieves better performances depending on the smoothness α of the solution u and the neural network, which is a significant advantage.

Results: Performance of PINNs Figure 3 shows the average relative error $\mathcal{L}$ using solid lines, with the maximum and minimum errors depicted by shaded areas. As N increases, the relative error $\mathcal{L}$ decreases and eventually reaches saturation. This saturation is attributed to several factors: the limitations in the network architecture as mentioned in Theorem 1, numerical errors in the datasets, discretization errors in relative error $\mathcal{L}$, and rounding errors in computation.

We report the minimum numbers N of collocation points with which the relative error $\mathcal{L}$ was saturated in the left half of Table 1. Specifically, we consider a relative error $\mathcal{L}$ below 130 % of the minimum observed one as saturated; the thresholds are denoted by horizontal red lines in Fig. 3. The proposed GLT exhibited competitive performances with considerably fewer collocation points. Specifically, it required less than half the number of points as compared to the second-best methods, and in the case of $s = 2$-dimensional Poisson's equation, it needed only a seventh of the points. These findings indicate that the proposed GLT can reduce computational costs significantly. Note that, for $s = 4$-dimensional Poisson's equation, the performance of the uniformly spaced sampling was extremely inferior, resulting in it not being captured in the image.

Following this, we standardized the number N of collocation points (and hence, the computational cost). In the right half of Table 1, we list the relative error $\mathcal{L}$ observed for collocation points where the relative error $\mathcal{L}$ of one of the comparison methods reached saturation, as denoted by vertical green lines in Fig. 3. A smaller error $\mathcal{L}$ indicates that a method outperforms others at the same computational cost. The proposed GLT yielded the smallest relative errors $\mathcal{L}$ across all cases, with a particularly pronounced difference in Poisson's equation with $s = 2$. We show the true solutions and the residuals of example results with such N in Appendix "Additional Results."

Therefore, we conclude that the proposed GLT can solve various PDEs with better performances and fewer collocation points, provided the dimension number s of the domain Ω is four or less—a range adequate for most physical simulations. We also confirmed that the periodization tricks significantly improve the overall performance in Ap-

pendix "Additional Results." Refer to Appendix "Higher Dimensional Case" for higher-dimensional cases.

Results: System Identification We further assessed the performance of white-box system identification, in a similar way as in those in Raissi et al. (2019). For the KdV and AC equations, we treated two parameters, (θ_1, θ_2), as learnable parameters $(\tilde{\theta}_1, \tilde{\theta}_2)$ and initialized them to zero. We extracted the true solutions u at N_s randomly selected points $\{\boldsymbol{x}_j\}_{j=0}^{N_s-1}$ as observations. During the training, in addition to the physics-informed loss, we minimized the mean squared error of the state $\tilde{u}$ at these points, that is, $\frac{1}{N_s}\sum_{j=0}^{N_s-1} \|u(\boldsymbol{x}_j) - \tilde{u}(\boldsymbol{x}_j)\|^2$. This procedure guides the learnable parameters $(\tilde{\theta}_1, \tilde{\theta}_2)$ to the true values (θ_1, θ_2). We set the number of points N_s to 100 for the KdV equation and 200 for the AC equation, which were nearly the minimum required for successful system identification. All other experimental settings were identical to those in the previous experiments.

Figure 4 shows the median of the relative error $\Delta\theta_1 = |\tilde{\theta}_1 - \theta_1|/|\theta_1|$ for the five trials. Our proposed GLT demonstrated the highest precision. Remarkably, it achieved comparable accuracy with approximately half the number of collocation points N required for the Sobol sequence in most cases, and significantly fewer than other methods. Recall that the observations $\{\boldsymbol{x}_j\}_{j=0}^{N_s-1}$ were selected using the Monte Carlo method, leading to the error in the order of $O(1/N_s^{\frac{1}{2}})$ for all methods. Nonetheless, the accuracy of parameter identification relies heavily on the strategy to determine the collocation points for the physics-informed loss and its number N. The proposed GLT method proved to be superior in this aspect as well.

Competitive Physics-Informed Neural Networks Competitive PINNs (CPINNs) are an improved version of PINNs with an additional neural network $D : \Omega \to \mathbb{R}$ called a discriminator (Zeng et al. 2023). Its objective function is $\frac{1}{N}\sum_{j=0}^{N-1} D(\boldsymbol{x}_j)\mathcal{N}[\tilde{u}](\boldsymbol{x}_j)$; the discriminator D is trained to maximize it, whereas the neural network $\tilde{u}$ is trained to minimize it, forming a zero-sum game. The Nash equilibrium offers the solution to a given PDE. CPINNs employed the competitive gradient descent algorithm to accelerate the convergence (Schaefer and Anandkumar 2019; Schaefer et al. 2020). The objective function is also regarded as a finite approximation to the integral $\int_{\boldsymbol{x}\in\Omega} D(\boldsymbol{x})\mathcal{N}[\tilde{u}](\boldsymbol{x})\mathrm{d}\boldsymbol{x}$; therefore, the proposed GLT in applicable to CPINNs.

We modified the code accompanying the manuscript and investigated the NLS and Burgers' equations[2]. The NLS equation is identical to the one above. See Appendix "Experimental Settings" for details about Burgers' equation. The number N of collocation points was 20,000 by default and varied. We folded the coordinates to ensure the periodicity of the loss function for the proposed GLT but did not ensure the initial and boundary conditions; we trained neural networks to learn the initial and boundary conditions following the original experimental settings. Also, we did not apply the randomization trick. All experiments were conducted using Python v3.9.16 and Pytorch v1.13.1 (Paszke et al. 2017) on servers with Intel Xeon Platinum 8368 and NVIDIA A100.

[2]See Supplementary Material at https://openreview.net/forum?id=z9SIj-IM7tn (MIT License)

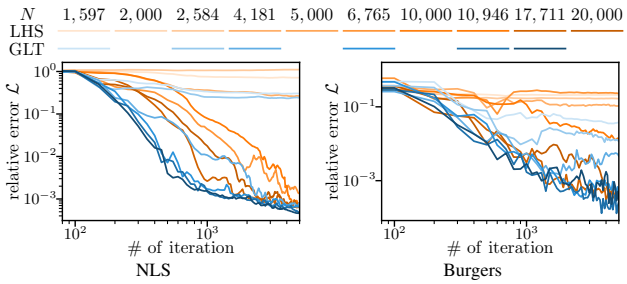

Figure 5: The results of CPINNs. The number of iterations and the relative error $\mathcal{L}$.

Results We have summarized the results in Fig. 5. In the case of the NLS equation, using LHS, the relative error $\mathcal{L}$ declines slowly even when $N = 20,000$, and there was almost no improvement for $N \leq 4,181$. Conversely, using GLT rapidly reduces the relative error $\mathcal{L}$ at $N = 6,765$. In most cases, GLT requires only one-third of the collocation points that LHS needs to achieve a comparable level of performance. In the case of the Burgers' equation, CPINNs using GLT demonstrated progress in learning with $N = 2,584$ collocation points, whereas CPINNs using LHS method with $N = 10,946$ achieved a worse performance rate. These results indicate that the proposed GLT exhibits competitive or superior convergence speed with 3 to 4 times fewer collocation points. The original paper demonstrated that CPINNs have a faster convergence rate than vanilla PINNs, but the GLT can further accelerate it.

Conclusion

This paper highlighted that the physics-informed loss, commonly used in PINNs and their variants, is a finite approximation to the integrated loss. From this perspective, we proposed good lattice training (GLT) to determine collocation points. This method enables a more accurate approximation of the integrated loss with a smaller number of collocation points. Experimental results using PINNs and CPINNs demonstrated that the GLT can achieve competitive or superior performance with much fewer collocation points. These results imply a significant reduction in computational cost and contribute to the large-scale computation of PINNs.

As shown in Figs. 3 and 5, the current problem setting reaches performance saturation with around $N = 1,000$ collocation points due to the network capacity and numerical errors in the datasets. However, Figure 2 demonstrates that the GLT significantly enhances the approximation accuracy even when using many more collocation points. This implies that the GLT is particularly effective in addressing larger-scale and high-precision problem settings, which will be explored further in future research.

Acknowledgments

This study was partially supported by JST CREST (JPMJCR1914), JST PRESTO (JPMJPR21C7), JST ASPIRE (JPMJAP2329), JST Moonshot R&D (JPMJMS2033-14), and JSPS KAKENHI (24K15105), and was achieved through the use of SQUID at D3 Center, Osaka University.

References

Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, G. S.; Davis, A.; Dean, J.; Devin, M.; Ghemawat, S.; Goodfellow, I.; Harp, A.; Irving, G.; Isard, M.; Jozefowicz, R.; Jia, Y.; Kaiser, L.; Kudlur, M.; Levenberg, J.; Mané, D.; Schuster, M.; Monga, R.; Moore, S.; Murray, D.; Olah, C.; Shlens, J.; Steiner, B.; Sutskever, I.; Talwar, K.; Tucker, P.; Vanhoucke, V.; Vasudevan, V.; Viégas, F.; Vinyals, O.; Warden, P.; Wattenberg, M.; Wicke, M.; Yu, Y.; and Zheng, X. 2016. TensorFlow: Large-scale Machine Learning on Heterogeneous Systems. *USENIX Symposium on Operating Systems Design and Implementation (OSDI)*.

Achdou, Y.; Buera, F. J.; Lasry, J.-M.; Lions, P.-L.; and Moll, B. 2014. Partial Differential Equation Models in Macroeconomics. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 372(2028): 20130397.

Bihlo, A.; and Popovych, R. O. 2022. Physics-Informed Neural Networks for the Shallow-Water Equations on the Sphere. *Journal of Computational Physics*, 456: 111024.

Chen, R. T. Q.; Rubanova, Y.; Bettencourt, J.; and Duvenaud, D. 2018. Neural Ordinary Differential Equations. In *Advances in Neural Information Processing Systems (NeurIPS)*, 1–19.

Chen, S.; Billings, S. A.; and Grant, P. M. 1990. Non-Linear System Identification Using Neural Networks. *International Journal of Control*, 51(6): 1191–1214.

Dissanayake, M. W. M. G.; and Phan-Thien, N. 1994. Neural-Network-Based Approximations for Solving Partial Differential Equations. *Communications in Numerical Methods in Engineering*, 10(3): 195–201.

Fang, K.-t.; and Wang, Y. 1994. *Number-Theoretic Methods in Statistics*. Series Title Monographs on Statistics and Applied Probability. Springer New York, NY.

Finzi, M.; Stanton, S.; Izmailov, P.; and Wilson, A. G. 2020. Generalizing Convolutional Neural Networks for Equivariance to Lie Groups on Arbitrary Continuous Data. In *International Conference on Machine Learning (ICML)*, 3146–3157. ISBN 978-1-71382-112-0.

Furihata, D.; and Matsuo, T. 2010. *Discrete Variational Derivative Method: A Structure-Preserving Numerical Method for Partial Differential Equations*. Chapman and Hall/CRC. ISBN 978-0-429-14846-0.

Greydanus, S.; Dzamba, M.; and Yosinski, J. 2019. Hamiltonian Neural Networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 1–16.

Gühring, I.; and Raslan, M. 2021. Approximation rates for neural networks with encodable weights in smoothness spaces. *Neural Networks*, 134: 107–130.

Hao, Z.; Ying, C.; Su, H.; Zhu, J.; Song, J.; and Cheng, Z. 2023. Bi-Level Physics-Informed Neural Networks for PDE Constrained Optimization Using Broyden's Hypergradients. In *International Conference on Learning Representations (ICLR)*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1–9. ISBN 978-1-4673-6964-0.

Heldmann, F.; Berkhahn, S.; Ehrhardt, M.; and Klamroth, K. 2023. PINN Training Using Biobjective Optimization: The Trade-off between Data Loss and Residual Loss. *Journal of Computational Physics*, 112211.

Hirsch, C. 2006. *Numerical Computation of Internal and External Flows*. Butterworth-Heinemann Limited. ISBN 978-0-7506-6595-7.

Jin, X.; Cai, S.; Li, H.; and Karniadakis, G. E. 2021. NSFnets (Navier-Stokes Flow Nets): Physics-informed Neural Networks for the Incompressible Navier-Stokes Equations. *Journal of Computational Physics*, 426: 109951.

Keng, H. L.; and Yuan, W. 1981. *Applications of Number Theory to Numerical Analysis*. Berlin, Heidelberg: Springer. ISBN 978-3-642-67831-8 978-3-642-67829-5.

Kingma, D. P.; and Ba, J. 2015. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations (ICLR)*, 1–15.

Knupp, P.; and Steinberg, S. 2020. *Fundamentals of Grid Generation*. CRC Press.

Korobov, N. M. 1959. The approximate computation of multiple integrals. In *Dokl. Akad. Nauk SSSR*, volume 124, 1207–1210. cir.nii.ac.jp.

Korobov, N. M. 1960. Properties and calculation of optimal coefficients. In *Dokl. Akad. Nauk SSSR*, volume 132, 1009–1012. mathnet.ru.

Krishnapriyan, A.; Gholami, A.; Zhe, S.; Kirby, R.; and Mahoney, M. W. 2022. Characterizing Possible Failure Modes in Physics-Informed Neural Networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 1–13.

Lagaris, I. E.; Likas, A.; and Fotiadis, D. I. 1998. Artificial Neural Networks for Solving Ordinary and Partial Differential Equations. *IEEE Transactions on Neural Networks*, 9(5): 987–1000.

Li, Z.; Zheng, H.; Kovachki, N.; Jin, D.; Chen, H.; Liu, B.; Azizzadenesheli, K.; and Anandkumar, A. 2021. Physics-Informed Neural Operator for Learning Partial Differential Equations.

Logan, J. D. 2015. *Applied Partial Differential Equations*. Undergraduate Texts in Mathematics. Cham: Springer Cham. ISBN 978-3-319-12492-6 978-3-319-12493-3.

Longo, M.; Mishra, S.; Rusch, T. K.; and Schwab, C. 2021. Higher-Order Quasi-Monte Carlo Training of Deep Neural Networks. *SIAM Journal on Scientific Computing*, 43(6): A3938–A3966.

Loshchilov, I.; and Hutter, F. 2017. SGDR: Stochastic Gradient Descent with Warm Restarts. In *International Conference on Learning Representations (ICLR)*, 1–16.

Lu, L.; Jin, P.; and Karniadakis, G. E. 2019. DeepONet: Learning Nonlinear Operators for Identifying Differential Equations Based on the Universal Approximation Theorem of Operators. *arXiv*, 3(3): 218–229.

Lu, Y.; Blanchet, J.; and Ying, L. 2022. Sobolev Acceleration and Statistical Optimality for Learning Elliptic Equations via Gradient Descent. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Lye, K. O.; Mishra, S.; and Ray, D. 2020. Deep Learning Observables in Computational Fluid Dynamics. *Journal of Computational Physics*, 410.

Mao, Z.; Jagtap, A. D.; and Karniadakis, G. E. 2020. Physics-Informed Neural Networks for High-Speed Flows. *Computer Methods in Applied Mechanics and Engineering*, 360: 112789.

Matsubara, T.; and Yaguchi, T. 2023. FINDE: Neural Differential Equations for Finding and Preserving Invariant Quantities. In *International Conference on Learning Representations (ICLR)*.

Mishra, S.; and Molinaro, R. 2021. Physics Informed Neural Networks for Simulating Radiative Transfer. *Journal of Quantitative Spectroscopy and Radiative Transfer*, 270: 107705.

Mishra, S.; and Molinaro, R. 2023. Estimates on the generalization error of Physics Informed Neural Networks (PINNs) for approximating PDEs. arXiv:2006.16144.

Morton, K. W.; and Mayers, D. F. 2005. *Numerical Solution of Partial Differential Equations: An Introduction*. Cambridge: Cambridge University Press, second edition. ISBN 978-0-521-60793-3.

Niederreiter, H. 1992. *Random Number Generation and Quasi-Monte Carlo Methods*. CBMS-NSF Regional Conference Series in Applied Mathematics. Society for Industrial and Applied Mathematics.

Paszke, A.; Chanan, G.; Lin, Z.; Gross, S.; Yang, E.; Antiga, L.; and Devito, Z. 2017. Automatic Differentiation in PyTorch. In *NeurIPS2017 Workshop on Autodiff*, 1–4.

Pokkunuru, A.; Rooshenas, P.; Strauss, T.; Abhishek, A.; and Khan, T. 2023. Improved Training of Physics-Informed Neural Networks Using Energy-Based Priors: A Study on Electrical Impedance Tomography. In *International Conference on Learning Representations (ICLR)*.

Raissi, M.; Perdikaris, P.; and Karniadakis, G. E. 2019. 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.

Rosofsky, S. G.; Majed, H. A.; and Huerta, E. A. 2023. Applications of Physics Informed Neural Operators. *Machine Learning: Science and Technology*, 4(2): 025022.

Schaefer, F.; and Anandkumar, A. 2019. Competitive Gradient Descent. In *Advances in Neural Information Processing Systems (NeurIPS)*, 1–11.

Schaefer, F.; Zheng, H.; and Anandkumar, A. 2020. Implicit Competitive Regularization in GANs. In *International Conference on Machine Learning (ICML)*, 8533–8544. PMLR.

Shankar, V.; Kirby, R. M.; and Fogelson, A. L. 2018. Robust Node Generation for Mesh-free Discretizations on Irregular Domains and Surfaces. *SIAM Journal on Scientific Computing*, 40(4): A2584–A2608.

Sharma, R.; and Shankar, V. 2022. Accelerated Training of Physics-Informed Neural Networks (PINNs) Using Meshless Discretizations. In *Advances in Neural Information Processing Systems*.

Sloan, I. H.; and Joe, S. 1994. *Lattice Methods for Multiple Integration*. Clarendon Press.

Thomas, J. W. 1995. *Numerical Partial Differential Equations: Finite Difference Methods*, volume 22 of *Texts in Applied Mathematics*. New York, NY: Springer. ISBN 978-1-4419-3105-4 978-1-4899-7278-1.

Thompson, J. F.; Warsi, Z. U. A.; and Mastin, C. W. 1985. *Numerical grid generation: foundations and applications*. USA: Elsevier North-Holland, Inc.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention Is All You Need. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Wang, C.; Li, S.; He, D.; and Wang, L. 2022. Is $L\hat{2}$ Physics Informed Loss Always Suitable for Training Physics Informed Neural Network? In *Advances in Neural Information Processing Systems (NeurIPS)*.

Wang, S.; and Perdikaris, P. 2023. Long-Time Integration of Parametric Evolution Equations with Physics-Informed DeepONets. *Journal of Computational Physics*, 475: 111855.

Wang, S.; Teng, Y.; and Perdikaris, P. 2021. Understanding and Mitigating Gradient Flow Pathologies in Physics-Informed Neural Networks. *SIAM Journal on Scientific Computing*, 43(5): A3055–A3081.

Wang, S.; Yu, X.; and Perdikaris, P. 2022. When and Why PINNs Fail to Train: A Neural Tangent Kernel Perspective. *Journal of Computational Physics*, 449: 110768.

Wang, Y. J.; and Lin, C. T. 1998. Runge-Kutta Neural Network for Identification of Dynamical Systems in High Accuracy. *IEEE Transactions on Neural Networks*, 9(2): 294–307.

Zaremba, S. K. 1972. *Applications of Number Theory to Numerical Analysis*. Academic Press.

Zeng, Q.; Kothari, Y.; Bryngelson, S. H.; and Schaefer, F. T. 2023. Competitive Physics Informed Networks. In *International Conference on Learning Representations (ICLR)*, 1–12.