Journal of Scientific Computing (2022) 92:88
https://doi.org/10.1007/s10915-022-01939-z

Scientific Machine Learning Through Physics–Informed Neural Networks: Where we are and What's Next

Salvatore Cuomo[1] · Vincenzo Schiano Di Cola[2] · Fabio Giampaolo[1] · Gianluigi Rozza[3] · Maziar Raissi[4] · Francesco Piccialli[1]

Received: 14 January 2022 / Revised: 6 July 2022 / Accepted: 6 July 2022 /
Published online: 26 July 2022

Abstract
Physics-Informed Neural Networks (PINN) are neural networks (NNs) that encode model equations, like Partial Differential Equations (PDE), as a component of the neural network itself. PINNs are nowadays used to solve PDEs, fractional equations, integral-differential equations, and stochastic PDEs. This novel methodology has arisen as a multi-task learning framework in which a NN must fit observed data while reducing a PDE residual. This article provides a comprehensive review of the literature on PINNs: while the primary goal of the study was to characterize these networks and their related advantages and disadvantages. The review also attempts to incorporate publications on a broader range of collocation-based physics informed neural networks, which stars form the vanilla PINN, as well as many other variants, such as physics-constrained neural networks (PCNN), variational hp-VPINN, and conservative PINN (CPINN). The study indicates that most research has focused on customizing the PINN through different activation functions, gradient optimization techniques, neural network structures, and loss function structures. Despite the wide range of applications for which PINNs have been used, by demonstrating their ability to be more feasible in some contexts than classical numerical techniques like Finite Element Method (FEM), advancements are still possible, most notably theoretical issues that remain unresolved.

Keywords Physics–Informed Neural Networks · Scientific Machine Learning · Deep Neural Networks · Nonlinear equations · Numerical methods · Partial Differential Equations · Uncertainty

S. Cuomo, V. Schiano, F. Giampaolo, G. Rozza and M. Raissi are contributed equally to this study

✉ Francesco Piccialli
francesco.piccialli@unina.it

Extended author information available on the last page of the article

科学计算杂志 (2022) 92:88
https://doi.org/10.1007/s10915-022-01939-z

通过物理信息神经网络的科学机器学习：我们所在之处与未来方向

萨尔瓦托雷·库莫[1] ·文森佐·斯卡尼奥·迪科拉[2] ·法比奥·吉安帕奥洛[1] ·
吉安卢基·罗扎[3] ·马齐尔·拉西[4] ·弗朗切斯科·皮奇亚利1

收到：2022年1月14日/ 修订版在线发布ised: 6 July 2022 / Accepted: 6 July 2022 /
2022年7月26日

摘要
物理信息神经网络（PINN）是编码模型方程（如偏微分方程（PDE））作为神经网络本身一部分的神经网络（NNs）。PINNs如今被用于求解偏微分方程、分数方程、积分微分方程和随机偏微分方程。这种新颖的方法已成为一个多任务学习框架，其中神经网络必须拟合观测数据同时减少偏微分方程残差。本文对PINNs的文献进行了全面综述：研究的主要目标是为这些网络及其相关优缺点进行特征描述。综述还试图包含基于配置的物理信息神经网络的出版物，这些网络从基础物理信息神经网络（vanilla PINN）开始，以及许多其他变体，如物理约束神经网络（PCNN）、变分hp-VPINN和保守PINN（CPINN）。研究表明，大多数研究都集中在通过不同的激活函数、梯度优化技术、神经网络结构和损失函数结构来定制PINN。尽管PINNs已被用于广泛的应用，通过展示它们在某些情况下比经典的数值技术（如有限元方法（FEM））更具可行性，但仍有进步空间，最显著的是仍存在未解决的问题的理论问题。

关键词 物理信息神经网络 · 科学机器学习 · 深度神经网络 · 非线性方程 · 数值方法 · 偏微分方程 · 不确定性

S. Cuomo, V. Schiano, F. Giampaolo, G. Rozza 和 M. Raissi 对这项研究贡献相同

✉ Francesco Picciallifrancesco.piccialli@unina.it 扩展作者信息可在文章最后一页找到

1 Introduction

Deep neural networks have succeeded in tasks such as computer vision, natural language processing, and game theory. Deep Learning (DL) has transformed how categorization, pattern recognition, and regression tasks are performed across various application domains.

Deep neural networks are increasingly being used to tackle classical applied mathematics problems such as partial differential equations (PDEs) utilizing machine learning and artificial intelligence approaches. Due to, for example, significant nonlinearities, convection dominance, or shocks, some PDEs are notoriously difficult to solve using standard numerical approaches. Deep learning has recently emerged as a new paradigm of scientific computing thanks to neural networks' universal approximation and great expressivity. Recent studies have shown deep learning to be a promising method for building meta-models for fast predictions of dynamic systems. In particular, NNs have proven to represent the underlying nonlinear input-output relationship in complex systems. Unfortunately, dealing with such high dimensional-complex systems are not exempt from the curse of dimensionality, which Bellman first described in the context of optimal control problems [15]. However, machine learning-based algorithms are promising for solving PDEs [19].

Indeed, Blechschmidt and Ernst [19] consider machine learning-based PDE solution approaches will continue to be an important study subject in the next years as deep learning develops in methodological, theoretical, and algorithmic developments. Simple neural network models, such as MLPs with few hidden layers, were used in early work for solving differential equations [89]. Modern methods, based on NN techniques, take advantage of optimization frameworks and auto-differentiation, like Berg and Nyström [16] that suggested a unified deep neural network technique for estimating PDE solutions. Furthermore, it is envisioned that DNN will be able to create an interpretable hybrid Earth system model based on neural networks for Earth and climate sciences [68].

Nowadays, the literature does not have a clear nomenclature for integrating previous knowledge of a physical phenomenon with deep learning. 'Physics-informed,' 'physics-based,' 'physics-guided,' and 'theory-guided' are often some used terms. Kim et al [80] developed the overall taxonomy of what they called informed deep learning, followed by a literature review in the field of dynamical systems. Their taxonomy is organized into three conceptual stages: (i) what kind of deep neural network is used, (ii) how physical knowledge is represented, and (iii) how physical information is integrated. Inspired by their work, we will investigate PINNs, a 2017 framework, and demonstrate how neural network features are used, how physical information is supplied, and what physical problems have been solved in the literature.

1.1 What the PINNs are

Physics–Informed Neural Networks (PINNs) are a scientific machine learning technique used to solve problems involving Partial Differential Equations (PDEs). PINNs approximate PDE solutions by training a neural network to minimize a loss function; it includes terms reflecting the initial and boundary conditions along the space-time domain's boundary and the PDE residual at selected points in the domain (called collocation point). PINNs are deep-learning networks that, given an input point in the integration domain, produce an estimated solution in that point of a differential equation after training. Incorporating a residual network that encodes the governing physics equations is a significant novelty with PINNs. The basic concept behind PINN training is that it can be thought of as an unsupervised strategy that

1 引言

深度神经网络在计算机视觉、自然语言处理和博弈论等任务中取得了成功。深度学习（DL）已经改变了分类、模式识别和回归任务在各个应用领域中的执行方式。

深度神经网络越来越多地被用于解决利用机器学习和人工智能方法的偏微分方程（PDEs）等经典应用数学问题。由于例如显著的非线性、对流主导或冲击，一些PDEs使用标准数值方法难以求解。由于神经网络的通用逼近能力和强大的表达能力，深度学习最近已成为科学计算的一种新范式。最近的研究表明，深度学习是构建元模型以快速预测动态系统的有前景的方法。特别是，神经网络已被证明可以表示复杂系统中的底层非线性输入-输出关系。不幸的是，处理此类高维复杂系统不可避免地会受到维数灾难的影响，贝尔曼首先在最优控制问题的背景下描述了这一点 [15]。然而，基于机器学习的算法在求解PDEs [19]方面具有前景。

确实，Blechschmidt和Ernst [19] 认为，随着深度学习在方法论、理论和算法发展上的进步，基于机器学习的偏微分方程求解方法将继续成为未来几年重要的研究课题。早期工作中，用于求解微分方程 [89]的简单神经网络模型，如具有少量隐藏层的多层感知机（MLPs），被使用。现代方法基于神经网络技术，利用优化框架和自动微分，如Berg和Nyström [16] 提出的统一深度神经网络技术，用于估计偏微分方程解。此外，人们设想深度神经网络（DNN）将能够基于神经网络创建一个可解释的混合地球系统模型，用于地球和气候科学 [68]。

如今，文献中还没有一个明确的术语来整合物理现象的先前知识与深度学习。'物理信息融合'、'物理基础'、'物理引导的'和'理论引导的'是常用的一些术语。金等人 [80]开发了他们所说的信息融合深度学习的整体分类法，并随后在动力系统领域进行了文献综述。他们的分类法分为三个概念阶段：(i) 使用了什么样的深度神经网络，(ii) 物理知识如何表示，以及(iii) 物理信息如何整合。受其工作的启发，我们将研究物理信息神经网络（PINNs），这是一个2017年的框架，并展示神经网络特征如何被使用，物理信息如何被提供，以及文献中已经解决了哪些物理问题。

1.1 What the PINNs are

物理信息神经网络（PINNs）是一种科学机器学习技术，用于解决涉及偏微分方程（PDEs）的问题。PINNs通过训练神经网络来逼近偏微分方程解，该神经网络用于最小化一个损失函数；该损失函数包含反映初始和边界条件的项，这些条件位于时空域的边界上，以及偏微分方程残差，该残差在域中的选定点（称为配置点）处计算。PINNs是深度学习网络，给定积分域中的一个输入点，在训练后在该点生成微分方程的估计解。将编码控制物理方程的残差网络纳入PINNs是一项重要创新。PINN训练的基本概念是，可以将其视为一种无监督策略，

does not require labelled data, such as results from prior simulations or experiments. The PINN algorithm is essentially a mesh-free technique that finds PDE solutions by converting the problem of directly solving the governing equations into a loss function optimization problem. It works by integrating the mathematical model into the network and reinforcing the loss function with a residual term from the governing equation, which acts as a penalizing term to restrict the space of acceptable solutions.

PINNs take into account the underlying PDE, i.e. the physics of the problem, rather than attempting to deduce the solution based solely on data, i.e. by fitting a neural network to a set of state-value pairs. The idea of creating physics-informed learning machines that employ systematically structured prior knowledge about the solution can be traced back to earlier research by Owhadi [125], which revealed the promising technique of leveraging such prior knowledge. Raissi et al [141, 142] used Gaussian process regression to construct representations of linear operator functionals, accurately inferring the solution and providing uncertainty estimates for a variety of physical problems; this was then extended in [140, 145]. PINNs were introduced in 2017 as a new class of data-driven solvers in a two-part article [143, 144] published in a merged version afterwards in 2019 [146]. Raissi et al [146] introduce and illustrate the PINN approach for solving nonlinear PDEs, like Schrödinger, Burgers, and Allen–Cahn equations. They created physics-informed neural networks (PINNs) which can handle both forward problems of estimating the solutions of governing mathematical models and inverse problems, where the model parameters are learnt from observable data.

The concept of incorporating prior knowledge into a machine learning algorithm is not entirely novel. In fact Dissanayake and Phan-Thien [39] can be considered one of the first PINNs. This paper followed the results of the universal approximation achievements of the late 1980s, [65]; then in the early 90s several methodologies were proposed to use neural networks to approximate PDEs, like the work on constrained neural networks [89, 135] or [93]. In particular Dissanayake and Phan-Thien [39] employed a simple neural networks to approximate a PDE, where the neural network's output was a single value that was designed to approximate the solution value at the specified input position. The network had two hidden layers, with 3, 5 or 10 nodes for each layer. The network loss function approximated the L^2 error of the approximation on the interior and boundary of the domain using point-collocation. While, the loss is evaluated using a quasi-Newtonian approach and the gradients are evaluated using finite difference.

In Lagaris et al [89], the solution of a differential equation is expressed as a constant term and an adjustable term with unknown parameters, the best parameter values are determined via a neural network. However, their method only works for problems with regular borders. Lagaris et al [90] extends the method to problems with irregular borders.

As computing power increased during the 2000s, increasingly sophisticated models with more parameters and numerous layers became the standard Özbay et al [127]. Different deep model using MLP, were introduced, also using Radial Basis Function Kumar and Yadav [87].

Research into the use of NNs to solve PDEs continued to gain traction in the late 2010s, thanks to advancements in the hardware used to run NN training, the discovery of better practices for training NNs, and the availability of open-source packages, such as Tensorflow [55], and the availability of Automatic differentiation in such packages [129].

Finally, more recent advancements by Kondor and Trivedi [83], and Mallat [102], brought to Raissi et al [146] solution that extended previous notions while also introducing fundamentally new approaches, such as a discrete time-stepping scheme that efficiently leverages the predictive ability of neural networks [82]. The fact that the framework could be utilized directly by plugging it into any differential problem simplified the learning curve for beginning to use such, and it was the first step for many researchers who wanted to solve their

它不需要标记数据，例如来自先前模拟或实验的结果。PINN算法本质上是一种无网格技术，通过将直接求解控制方程的问题转换为损失函数优化问题来找到偏微分方程解。它通过将数学模型集成到网络中，并使用来自控制方程的残差项来加强损失函数，该残差项作为惩罚项来限制可接受的解空间。

PINNs考虑了底层偏微分方程，即问题的物理原理，而不是试图仅基于数据推导出解，即通过将神经网络拟合到一组状态-值对。创建物理信息学习机器的想法，即利用关于解的系统结构化先验知识，可以追溯到Owhadi [125], 的早期研究，该研究揭示了利用这种先验知识的有前景的技术。Raissi等人 [141, 142]使用高斯过程回归来构建线性算子泛函的表示，准确推断解并为各种物理问题提供不确定性估计；这随后在 [140, 145]中得到了扩展。PINNs于2017年作为一类新的数据驱动求解器被引入，发表在后续合并版本的两篇文章 [143,144] 中。Raissi等人 [146] 介绍并说明了PINN方法用于求解非线性偏微分方程，如薛定谔方程、布格斯方程和艾伦-卡恩方程。他们创建了物理信息神经网络（PINNs），这些网络可以处理估计控制数学模型解的正向问题，以及模型参数从可观察数据中学习的逆向问题。

将先验知识融入机器学习算法的概念并非全新。事实上，迪桑纳亚克和范天 [39] 可以被认为是首批物理信息神经网络 (PINNs)。本文遵循了 20 世纪 80 年代末通用逼近的成果，[65]；然后在 90 年代初，提出了几种使用神经网络逼近偏微分方程 (PDEs) 的方法，例如关于约束神经网络 [89, 135] 或 [93] 的研究。特别是，迪桑纳亚克和范天 [39] 采用简单的神经网络来逼近一个偏微分方程，其中神经网络的输出是一个单一值，该值被设计为逼近在指定输入位置处的解值。该网络有两个隐藏层，每层有 3、5 或 10 个节点。网络损失函数使用点配置法在域的内部和边界上逼近逼近的 $L2$ 误差。然而，损失值是使用拟牛顿法评估的，梯度是使用有限差分法评估的。

在拉加里斯等人 [89], 的研究中，微分方程的解被表示为一个常数项和一个具有未知参数的可调项，最佳参数值通过神经网络确定。然而，他们的方法仅适用于具有规则边界的 问题。拉加里斯等人 [90] 将该方法扩展到具有不规则边界的 问题。

随着2000年代计算能力的提升，具有更多参数和多层 increasingly sophisticated 模型成为标准。Özbay等人 [127]引入了不同的深度模型，使用 MLP，还使用了径向基函数 Kumar和Yadav [87]。

在2010年代后期，使用神经网络NNs解决偏微分方程PDEs的研究持续获得关注，这得益于用于运行神经网络训练的硬件的进步、训练神经网络更好的实践以及开源软件包的可用性，例如Tensorflow[55], 以及这些软件包中可用的自动微分 [129]。

最后，Kondor和Trivedi [83], 以及Mallat [102], 的最新进展为Raissi等人 [146] 带来了扩展了先前概念的同时也引入了根本性新方法的解决方案，例如一个离散时间步长方案，该方案有效地利用了神经网络的预测能力 [82]。该框架可以直接通过将其插入任何微分问题来使用，简化了开始使用此类问题的学习曲线，对于许多希望解决他们

problems with a Neural network approach [105]. The success of the PINNs can be seen from the rate at which Raissi et al [146] is cited, and the exponentially growing number of citations in the recent years (Fig. 1).

However, PINN is not the only NN framework utilized to solve PDEs. Various frameworks have been proposed in recent years, and, while not exhaustive, we have attempted to highlight the most relevant ones in this paragraph.

The Deep Ritz method (DRM) [42], where the loss is defined as the energy of the problem's solution.

Then there approaches based on the Galerkin method, or Petrov–Galerkin method, where the loss is given by multiplying the residual by a test function, and when is the volumetric residual we have a Deep Galerkin Method (DGM) [160]. Whereas, when a Galerkin approach is used on collocation points the framework is a variant of PINNs, i.e. a hp-VPINN Kharazmi et al [78].

Within the a collocation based approach, i.e. PINN methodology [109, 146, 191], many other variants were proposed, as the variational hp-VPINN, as well as conservative PINN (CPINN)[71]. Another approach is physics-constrained neural networks (PCNNs) [97, 168, 200]. while PINNs incorporate both the PDE and its initial/boundary conditions (soft BC) in the training loss function, PCNNs, are "data-free" NNs, i.e. they enforce the initial/boundary conditions (hard BC) via a custom NN architecture while embedding the PDE in the training loss. This soft form technique is described in Raissi et al [146], where the term "physics-informed neural networks" was coined (PINNs).

Because there are more articles on PINN than any other specific variant, such as PCNN, hp-VPINN, CPINN, and so on, this review will primarily focus on PINN, with some discussion of the various variants that have emerged, that is, NN architectures that solve differential equations based on collocation points.

Finally, the acronym PINN will be written in its singular form rather than its plural form in this review, as it is considered representative of a family of neural networks of the same type.

Various review papers involving PINN have been published. About the potentials, limitations and applications for forward and inverse problems [74] for three-dimensional flows [20], or a comparison with other ML techniques [19]. An introductory course on PINNs that covers the fundamentals of Machine Learning and Neural Networks can be found from Kollmannsberger et al [82]. PINN is also compared against other methods that can be applied to solve PDEs, like the one based on the Feynman–Kac theorem [19]. Finally, PINNs have also been extended to solve integrodifferential equations (IDEs) [128, 193] or stochastic differential equations (SDEs) [189, 195].

Being able to learn PDEs, PINNs have several advantages over conventional methods. PINNs, in particular, are mesh-free methods that enable on-demand solution computation after a training stage, and they allow solutions to be made differentiable using analytical gradients. Finally, they provide an easy way to solve forward jointly and inverse problems using the same optimization problem. In addition to solving differential equations (the forward problem), PINNs may be used to solve inverse problems such as characterizing fluid flows from sensor data. In fact, the same code used to solve forward problems can be used to solve inverse problems with minimal modification. Moreover, in the context of inverse design, PDEs can also be enforced as hard constraints (hPINN) [101]. Indeed, PINNs can address PDEs in domains with very complicated geometries or in very high dimensions that are all difficult to numerically simulate as well as inverse problems and constrained optimization problems.

与神经网络方法相关的问题 [105]。PINNs的成功可以从Raissi等人 [146] 被引用的频率以及近年来呈指数增长的引用数量（图 1）。

然而，PINN并非唯一用于求解偏微分方程（PDEs）的神经网络框架。近年来提出了各种框架，虽然不全面，但我们试图在本段中突出最相关的一些框架。

深度里茨方法（DRM）[42], ，其中损失定义为问题解的能量。

然后有一些基于伽辽金方法或佩特罗夫-伽辽金方法的方法，其中损失由残差乘以一个测试函数给出，当使用体积残差时，我们有一个深度伽辽金方法（DGM）[160]。而，当在配置点使用伽辽金方法时，该框架是PINNs的一个变体，即hp-VPINN Kharazmi等人 [78]。

在基于配置点的方法中，即PINN方法学 [109, 146, 191], ，提出了许多其他变体，如变分hp-VPINN，以及保守PINN（CPINN）[71]。另一种方法是物理约束神经网络（PCNNs）[97, 168,200]。虽然PINNs将偏微分方程及其初始/边界条件（软边界条件）纳入训练损失函数，但PCNNs是"无数据"的神经网络，即它们通过自定义NN架构强制执行初始/边界条件（硬边界条件），同时在训练损失中嵌入偏微分方程。这种软形式技术由Raissi等人 [146], 描述，其中首次提出了"物理信息神经网络"这一术语（PINNs）。

由于关于PINN的文章比任何其他特定变体（如PCNN、hp-VPINN、CPINN等）都更多，因此本次综述将主要关注PINN，并对出现的各种变体进行一些讨论，即基于配置点求解微分方程的NN架构。

最后，在本综述中，PINN缩写将以其单数形式而不是复数形式书写，因为它被认为是同类型神经网络家族的代表性代表。

已发表多篇涉及PINN的综述论文。关于三维流动[20], 的正向和逆向问题的潜力、局限性和应用 [74]，以及其他机器学习技术 [19]的比较。可以从Kollmannsberger等人 [82]处找到一本涵盖机器学习和神经网络基础的PINNs入门课程。PINN也被与其他可以应用于求解偏微分方程（PDEs）的方法进行了比较，例如基于费曼-卡克定理 [19]的方法。最后，PINNs已被扩展用于求解积分微分方程（IDEs） [128, 193] 或随机微分方程（SDEs） [189, 195]。

能够学习偏微分方程（PDEs），PINNs在传统方法上具有 several 优势。PINNs，特别是无网格方法，能够在训练阶段后进行按需求解计算，并允许使用解析梯度使解可微。最后，它们提供了一种简单的方法，可以使用同一个优化问题来联合求解正向和逆向问题。除了求解微分方程（正向问题），PINNs还可以用于求解逆问题，例如从传感器数据中表征流体流动。事实上，用于求解正向问题的相同代码可以用于求解逆问题，只需进行最小修改。此外，在逆向设计的背景下，偏微分方程（PDEs）也可以作为硬约束（hPINN）[101]进行施加。实际上，PINNs可以处理具有非常复杂几何形状或非常高维度的域中的偏微分方程（PDEs），这些域都难以数值模拟，以及逆问题和约束优化问题。

1.2 What is this Review About

In this survey, we focus on how PINNs are used to address different scientific computing problems, the building blocks of PINNs, the aspects related to learning theory, what toolsets are already available, future directions and recent trends, and issues regarding accuracy and convergence. According to different problems, we show how PINNs solvers have been customized in literature by configuring the depth, layer size, activation functions and using transfer learning.

This article can be considered as an extensive literature review on PINNs. It was mostly conducted by searching Scopus for the terms:

```
((physic* OR physical) W/2 (informed OR constrained) W/2
"neural network")
```

The primary research question was to determine what PINNs are and their associated benefits and drawbacks. The research also focused on the outputs from the CRUNCH research group in the Division of Applied Mathematics at Brown University and then on the (Physics–Informed Learning Machines for Multiscale and Multiphysics Problems) PhILMs Center, which is a collaboration with the Pacific Northwest National Laboratory. In such a frame, the primary authors who have been involved in this literature research are Karniadakis G.E., Perdikaris P., and Raissi M. Additionally, the review considered studies that addressed a broader topic than PINNs, namely physics-guided neural networks, as well as physics-informed machine learning and deep learning.

Figure 1 summarizes what the influence of PINN is in today's literature and applications.

2 The Building Blocks of a PINN

Physically-informed neural networks can address problems that are described by few data, or noisy experiment observations. Because they can use known data while adhering to any given physical law specified by general nonlinear partial differential equations, PINNs can also be considered neural networks that deal with supervised learning problems [52]. PINNs can solve differential equations expressed, in the most general form, like:

$$\begin{aligned} \mathcal{F}(u(z);\gamma) &= f(z) \qquad z \text{ in } \Omega, \\ \mathcal{B}(u(z)) &= g(z) \qquad z \text{ in } \partial\Omega \end{aligned} \tag{1}$$

defined on the domain $\Omega \subset \mathbb{R}^d$ with the boundary $\partial\Omega$. Where $z := [x_1, \ldots, x_{d-1}; t]$ indicates the space-time coordinate vector, u represents the unknown solution, γ are the parameters related to the physics, f is the function identifying the data of the problem and $\mathcal{F}$ is the non linear differential operator. Finally, since the initial condition can actually be considered as a type of Dirichlet boundary condition on the spatio-temporal domain, it is possible to denote $\mathcal{B}$ as the operator indicating arbitrary initial or boundary conditions related to the problem and g the boundary function. Indeed, the boundary conditions can be Dirichlet, Neumann, Robin, or periodic boundary conditions.

Equation (1) can describe numerous physical systems including both forward and inverse problems. The goal of forward problems is to find the function u for every z, where γ are specified parameters. In the case of the inverse problem, γ must also be determined from the data. The reader can find an operator based mathematical formulation of Eq. (1) in the work of Mishra and Molinaro [111].

1.2 本文研究内容

在本综述中，我们关注物理信息神经网络（PINNs）如何用于解决不同的科学计算问题、PINNs的构建块、与学习理论相关的内容、已有的工具集、未来方向和最新趋势，以及精度和收敛性问题。根据不同的问题，我们展示了文献中如何通过配置深度、层大小、激活函数和使用迁移学习来定制PINNs求解器。

本文可被视为关于物理信息神经网络（PINNs）的广泛文献综述。该综述主要通过在Scopus中搜索以下术语进行:

((物理* OR 物理的) W/2 (信息 OR 约束) W/2
"neural network")

主要研究问题在于确定物理信息神经网络（PINNs）是什么，以及它们的相关优势和缺点。研究还关注了布朗大学应用数学系CRUNCH研究小组的成果，然后是（用于多尺度和多物理场问题的物理信息学习机器）PhILMs中心，该中心是与太平洋西北国家实验室的合作项目。在这样的框架下，参与这项文献研究的主要作者有卡尼亚迪亚基斯 G.E.、佩德里卡里斯 P.和拉西 M.。此外，该综述还考虑了涉及比PINNs更广泛主题的研究，即物理引导神经网络，以及物理信息机器学习和深度学习。

图 1 总结了PINN在当今文献和应用中的影响。

2 PINN的构建块

物理信息神经网络可以解决由少量数据或噪声实验观测描述的问题。因为它们可以在遵循任何给定的、由一般非线性偏微分方程指定的物理定律的同时使用已知数据，所以PINNs也可以被视为处理监督学习问题的神经网络 [52]。PINNs可以求解以最一般形式表达的微分方程，如:

$$\begin{aligned} \mathcal{F}(u(z);\gamma) &= f(z) \qquad z \text{ in } \Omega, \\ \mathcal{B}(u(z)) &= g(z) \qquad z \text{ in } \partial\Omega \end{aligned} \tag{1}$$

定义在域 $\Omega \subset \mathbb{R}^d$ 上，具有边界条件 $\partial\Omega$。其中 $z := [x1, \ldots, x_{d-1}; t]$ 表示时空坐标向量，u 表示未知解，γ 是与物理相关的参数，f 是识别问题数据的函数，$\mathcal{F}$ 是非线性微分算子。最后，由于初始条件实际上可以被视为时空域上的一种狄利克雷边界条件，因此可以表示$\mathcal{B}$ 为表示与问题相关的任意初始或边界条件的算子，g 为边界函数。实际上，边界条件可以是狄利克雷、诺伊曼、罗宾或周期性边界条件。

方程（1）可以描述许多物理系统，包括正向和逆问题。正向问题的目标是找到对于每个 u 的函数 z，其中 γ 是指定的参数。在逆问题的情况下，γ 也必须从数据中确定。读者可以在Mishra和Molinaro [111]的工作中找到方程（1）的基于算子的数学公式。

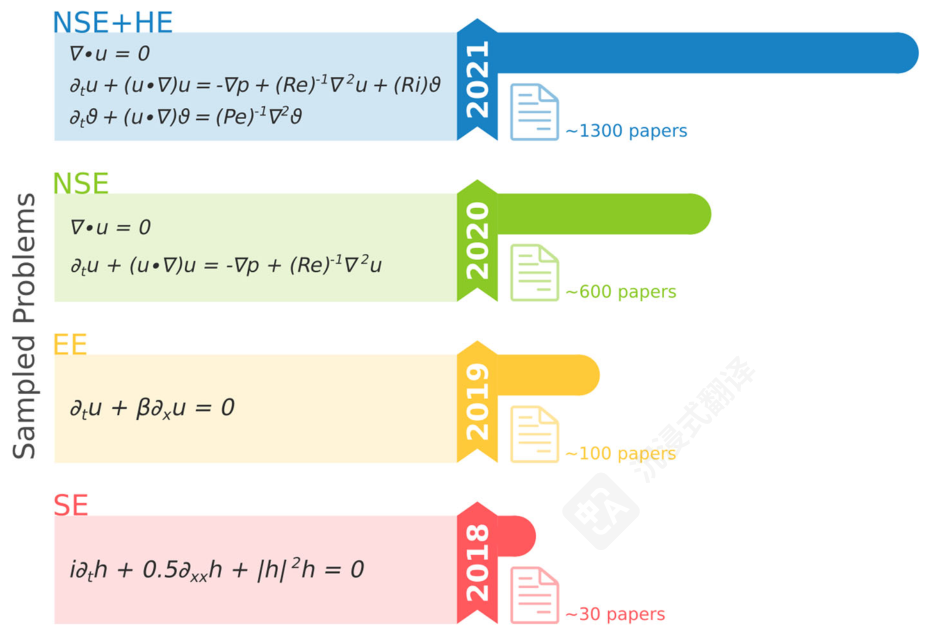

Fig. 1 A number of papers related to PINNs (on the right) addressed problems on which PINNs are applied (on the left) by year. PINN is having a significant impact in the literature and in scientific applications. The number of papers referencing Raissi or including PINN in their abstract title or keywords is increasing exponentially. The number of papers about PINN more than quintupled between 2019 and 2020, and there were twice as many new papers published in 2021 as there were in 2020. Since Raissi's first papers on arXiv in 2019 [146], a boost in citations can be seen in late 2018 and 2019. On the left, we display a sample problem solved in that year by one of the articles, specifically a type of time-dependent equation. Some of the addressed problems to be solved in the first vanilla PINN, for example, were the Allen–Cahn equation, the Korteweg–de Vries equation, or the 1D nonlinear Shrödinger problem (SE). By the end of 2019 Mao et al [103] solved with the same PINN Euler equations (EE) that model high-speed aerodynamic flows. By the end of 2020 Jin et al [73] solved the incompressible Navier–Stokes equations (NSE). Finally, in 2021 Cai et al [22] coupled the Navier–Stokes equations with the corresponding temperature equation for analyzing heat flow convection (NSE+HE)

In the PINN methodology, $u(z)$ is computationally predicted by a NN, parametrized by a set of parameters θ, giving rise to an approximation

$$\hat{u}_\theta(z) \approx u(z);$$

where $\hat{(\cdot)}_\theta$ denotes a NN approximation realized with θ.

In this context, where forward and inverse problems are analyzed in the same framework, and given that PINN can adequately solve both problems, we will use θ to represent both the vector of all unknown parameters in the neural network that represents the surrogate model and unknown parameters γ in the case of an inverse problem.

In such a context, the NN must learn to approximate the differential equations through finding θ that define the NN by minimizing a loss function that depends on the differential equation $\mathcal{L}_\mathcal{F}$, the boundary conditions $\mathcal{L}_\mathcal{B}$, and eventually some known data $\mathcal{L}_{data}$, each of them adequately weighted:

$$\theta^* = \arg\min_\theta \left(\omega_\mathcal{F}\mathcal{L}_\mathcal{F}(\theta) + \omega_\mathcal{B}\mathcal{L}_\mathcal{B}(\theta) + \omega_d\mathcal{L}_{data}(\theta)\right). \qquad (2)$$

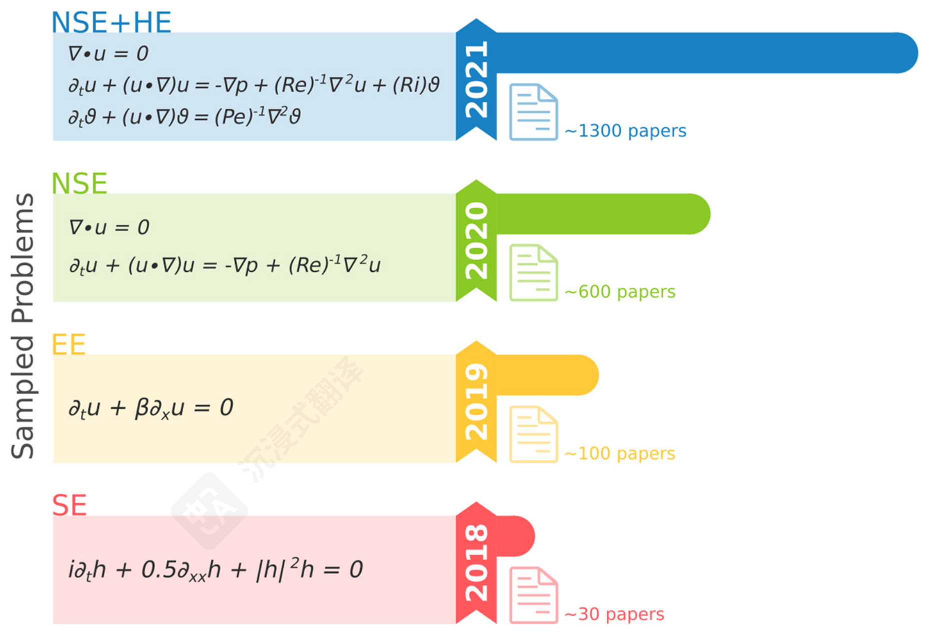

图1 许多与物理信息神经网络（PINNs）相关的论文（右侧）通过年份展示了PINNs应用的问题（左侧）。PINNs在文献和科学应用中具有重要影响。引用Raissi或在其摘要标题或关键词中包含PINN的论文数量呈指数增长。2019年至2020年，关于PINN的论文数量增加了五倍以上，2021年发表的论文数量是2020年的两倍。自Raissi在arXiv上首次发表论文以来 2019 [146],在2018年末和2019年末，引用量显著增加。在左侧，我们展示了一篇论文在那一年解决的样本问题，具体是一种时变方程。首批基础物理信息神经网络（vanilla PINN）解决的某些问题，例如艾伦-卡恩方程、科尔滕韦格-德弗里斯方程或一维非线性薛定谔问题（SE）。到2019年底，毛等人 [103] 使用相同的PINN解决了欧拉方程（EE），该方程模拟高速空气动力学流动。到2020年底，金等人 [73] 解决了不可压缩纳维-斯托克斯方程（NSE）。最后，在2021年，蔡等人 [22] 将纳维-斯托克斯方程与其相应的温度方程耦合起来，用于分析热流对流（NSE+HE）

在PINN方法中，$u(z)$ 由神经网络计算预测，该神经网络由一组参数 θ参数化，从而产生一个近似值

$$\hat{u}_\theta(z) \approx u(z);$$

其中 $\hat{(\cdot)}_\theta$ 表示使用 θ实现的神经网络近似

在这种情况下，正向和逆问题在同一个框架中进行分析，并且由于PINN可以充分解决这两个问题，我们将使用 θ 来表示表示代理模型的神经网络的未知参数向量和逆问题中的未知参数 γ

在这种情况下，神经网络必须学习通过找到定义神经网络并最小化依赖于微分方程$\mathcal{L}_\mathcal{F}$、边界条件 $\mathcal{L}_\mathcal{B}$以及最终一些已知数据 $\mathcal{L}_{数据}$的损失函数（每个都被充分加权）来近似微分方程：

$$\theta^* = \arg\min_\theta \left(\omega_\mathcal{F}\mathcal{L}_\mathcal{F}(\theta) + \omega_\mathcal{B}\mathcal{L}_\mathcal{B}(\theta) + \omega_d\mathcal{L}_{data}(\theta)\right). \qquad (2)$$

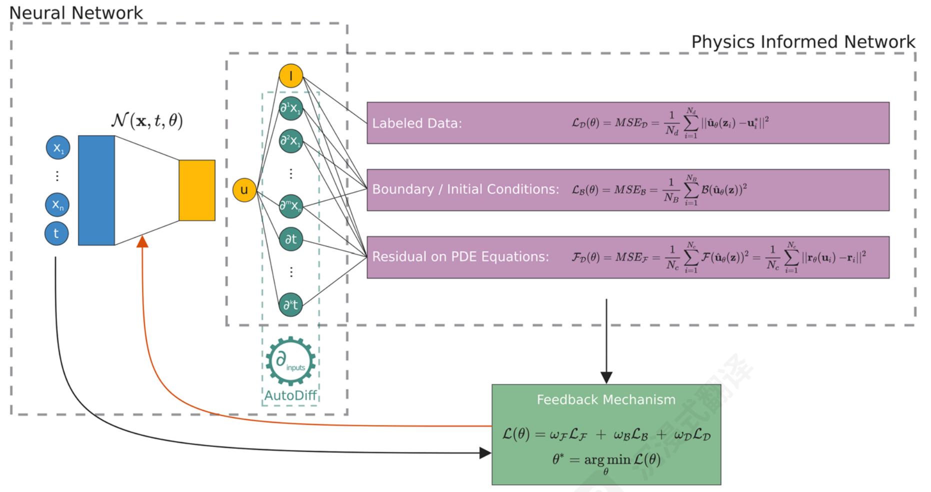
Neural Network
Physics Informed Network
$\mathcal{N}(\mathbf{x}, t, \theta)$
x₁
xₙ
t
u
I
∂¹x₁
∂²x₁
∂ᵐxₙ
∂t
∂ᵏt
inputs
AutoDiff
Labeled Data: $\mathcal{L}_{\mathcal{D}}(\theta) = MSE_{\mathcal{D}} = \frac{1}{N_d}\sum_{i=1}^{N_d}||\hat{\mathbf{u}}_\theta(\mathbf{z}_i) - \mathbf{u}_i^*||^2$
Boundary / Initial Conditions: $\mathcal{L}_{\mathcal{B}}(\theta) = MSE_{\mathcal{B}} = \frac{1}{N_B}\sum_{i=1}^{N_B}\mathcal{B}(\hat{\mathbf{u}}_\theta(\mathbf{z}))^2$
Residual on PDE Equations: $\mathcal{F}_{\mathcal{D}}(\theta) = MSE_{\mathcal{F}} = \frac{1}{N_c}\sum_{i=1}^{N_c}\mathcal{F}(\hat{\mathbf{u}}_\theta(\mathbf{z}))^2 = \frac{1}{N_c}\sum_{i=1}^{N_c}||\mathbf{r}_\theta(\mathbf{u}_i) - \mathbf{r}_i||^2$
Feedback Mechanism
$\mathcal{L}(\theta) = \omega_{\mathcal{F}}\mathcal{L}_{\mathcal{F}} + \omega_{\mathcal{B}}\mathcal{L}_{\mathcal{B}} + \omega_{\mathcal{D}}\mathcal{L}_{\mathcal{D}}$
$\theta^* = \arg\min_\theta \mathcal{L}(\theta)$

Fig. 2 Physics-informed neural networks building blocks. PINNs are made up of differential equation residual (loss) terms, as well as initial and boundary conditions. The network's inputs (variables) are transformed into network outputs (the field u). The network is defined by θ. The second area is the physics informed network, which takes the output field u and computes the derivative using the given equations. The boundary/initial condition is also evaluated (if it has not already been hard-encoded in the neural network), and also the labeled data observations are calculated (in case there is any data available). The final step with all of these residuals is the feedback mechanism, that minimizes the loss, using an optimizer, according to some learning rate in order to obtain the NN parameters θ

To summarize, PINN can be viewed as an unsupervised learning approach when they are trained solely using physical equations and boundary conditions for forward problems; however, for inverse problems or when some physical properties are derived from data that may be noisy, PINN can be considered supervised learning methodologies.

In the following paragraph, we will discuss the types of NN used to approximate $u(z)$, how the information derived by $\mathcal{F}$ is incorporated in the model, and how the NN learns from the equations and additional given data.

Figure 2 summarizes all the PINN's building blocks discussed in this section. PINN are composed of three components: a neural network, a physics-informed network, and a feedback mechanism. The first block is a NN, $\hat{u}_\theta$, that accepts vector variables z from the Eq. (1) and outputs the filed value u. The second block can be thought of PINN's functional component, as it computes the derivative to determine the losses of equation terms, as well as the terms of the initial and boundary conditions of Eq. (2). Generally, the first two blocks are linked using algorithmic differentiation, which is used to inject physical equations into the NN during the training phase. Thus, the feedback mechanism minimizes the loss according to some learning rate, in order to fix the NN parameters vector θ of the NN $\hat{u}_\theta$. In the following, it will be clear from the context to what network we are referring to, whether the NN or the functional network that derives the physical information.

2.1 Neural Network Architecture

The representational ability of neural networks is well established. According to the universal approximation theorem, any continuous function can be arbitrarily closely approximated by a multi-layer perceptron with only one hidden layer and a finite number of neurons [17, 34,

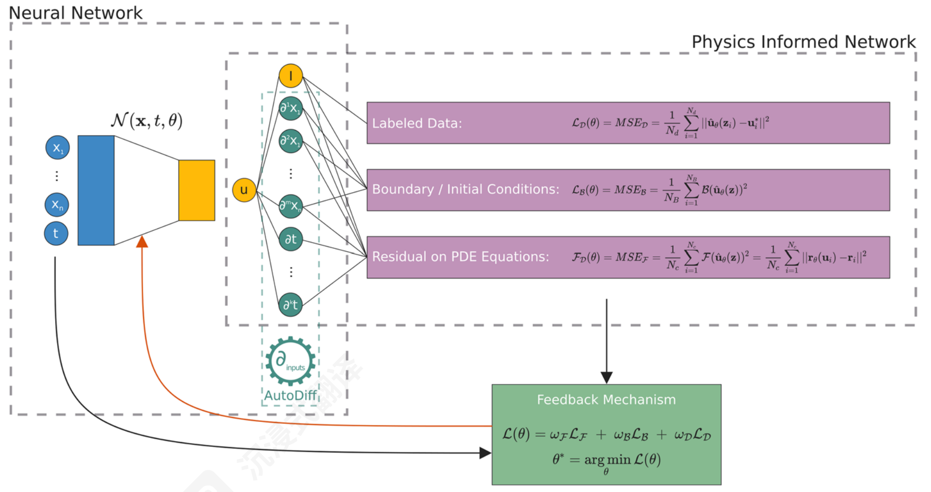
Neural Network
Physics Informed Network
$\mathcal{N}(\mathbf{x}, t, \theta)$
Labeled Data:
Boundary / Initial Conditions:
Residual on PDE Equations:
AutoDiff
Feedback Mechanism

Fig. 2 Physics-informed neural networks building blocks. PINNs are made up of differential equation residual (loss) terms, as well as initial and boundary conditions. The network's inputs (variables) are transformed into network outputs (the field u). The network is defined by θ. The second area is the physics informed network, which takes the output field u and computes the derivative using the given equations. The boundary/initial condition is also evaluated (if it has not already been hard-encoded in the neural network), and also the labeled data observations are calculated (in case there is any data available). The final step with all of these residuals is the feedback mechanism, that minimizes the loss, using an optimizer, according to some learning rate in order to obtain the NN parameters θ

总结来说，当PINN仅使用物理方程和正向问题的边界条件进行训练时，可以被视为一种无监督学习方法；然而，对于逆问题或当某些物理属性从可能存在噪声的数据中推导出来时，PINN可以被视为监督学习方法。

在下一段中，我们将讨论用于近似 $u(z)$ 的神经网络类型，如何将 <style id='7'>$\mathcal{F}$ 推导出的信息整合到模型中，以及神经网络如何从方程和额外给定数据中学习。

图 2 总结了本节中讨论的所有PINN构建块。PINN由三个组件组成：神经网络、物理信息网络和反馈机制。第一个构建块是一个神经网络， $\hat{}u_\theta$，它接受来自方程（1）的向量变量 z，并输出场值u。第二个构建块可以被视为PINN的功能组件，因为它计算导数以确定方程项的损失，以及方程（2）的初始和边界条件项。通常，前两个构建块通过算法微分连接，该微分在训练阶段将物理方程注入神经网络。因此，反馈机制根据某些学习率最小化损失，以固定神经网络参数向量 θ 的神经网络 $\hat{}u_\theta$。在下文中，从上下文将清楚我们指的是神经网络还是推导出物理信息的功能网络。

2.1 神经网络架构

神经网络的表征能力已经得到充分证实。根据通用逼近定理，任何连续函数都可以通过一个具有单个隐藏层和有限数量神经元的多层感知器任意紧密地逼近 [17, 34,

65, 192]. While neural networks can express very complex functions compactly, determining the precise parameters (weights and biases) required to solve a specific PDE can be difficult [175]. Furthermore, identifying the appropriate artificial neural network (ANN) architecture can be challenging. There are two approaches: shallow learning networks, which have a single hidden layer and can still produce any non-linear continuous function, and deep neural networks (DNN), a type of ANN that uses more than two layers of neural networks to model complicated relationships [2]. Finally, the taxonomy of various Deep Learning architectures is still a matter of research [2, 117, 155]

The main architectures covered in the Deep Learning literature include fully connected feed-forward networks (called FFNN, or also FNN, FCNN, or FF–DNN), convolutional neural networks (CNNs), and recurrent neural networks (RNNs) [29, 92]. Other more recent architectures encompass Auto–Encoder (AE), Deep Belief Network (DBN), Generative Adversarial Network (GAN), and Bayesian Deep Learning (BDL) [5, 17].

Various PINN extensions have been investigated, based on some of these networks. An example is estimating the PINN solution's uncertainty using Bayesian neural networks. Alternatively, when training CNNs and RNNs, finite-difference filters can be used to construct the differential equation residual in a PINN-like loss function. This section will illustrate all of these examples and more; first, we will define the mathematical framework for characterizing the various networks.

According to Caterini and Chang [25] a generic deep neural network with L layers, can be represented as the composition of L functions $f_i(x_i, \theta_i)$ where x_i are known as state variables, and θ_i denote the set of parameters for the i-th layer. So a function $u(x)$ is approximated as

$$u_\theta(x) = f_L \circ f_{L-1} \ldots \circ f_1(x). \tag{3}$$

where each f_i is defined on two inner product spaces E_i and H_i, being $f_i \in E_i \times H_i$ and the layer composition, represented by $\circ$, is to be read as $f_2 \circ f_1(x) = f_2(f_1(x))$.

Since Raissi et al [146] original vanilla PINN, the majority of solutions have used feed-forward neural networks. However, some researchers have experimented with different types of neural networks to see how they affect overall PINN performance, in particular with CNN, RNN, and GAN, and there seems there has been not yet a full investigation of other networks within the PINN framework. In this subsection, we attempt to synthesize the mainstream solutions, utilizing feed-forward networks, and all of the other versions. Table 1 contains a synthesis reporting the kind of neural network and the paper that implemented it.

2.1.1 Feed-forward Neural Network

A feed-forward neural network, also known as a multi-layer perceptron (MLP), is a collection of neurons organized in layers that perform calculations sequentially through the layers. Feed-forward networks, also known as multilayer perceptrons (MLP), are DNNs with several hidden layers that only move forward (no loopback). In a fully connected neural network, neurons in adjacent layers are connected, whereas neurons inside a single layer are not linked. Each neuron is a mathematical operation that applies an activation function to the weighted sum of it's own inputs plus a bias factor [175]. Given an input $x \in \Omega$ an MLP transforms it to an output, through a layer of units (neurons) which compose of affine-linear maps between units (in successive layers) and scalar non-linear activation functions within units, resulting in a composition of functions. So for MLP, it is possible to specify (3) as

$$f_i(x_i; W_i, b_i) = \alpha_i(W_i \cdot x_i + b_i)$$

65, 192]. 虽然神经网络可以紧凑地表达非常复杂的函数，但要确定解决特定PDE所需的精确参数（权重和偏差）可能很困难[175]。此外，确定合适的人工神经网络（ANN）架构可能具有挑战性。有两种方法：浅层学习网络，它具有单个隐藏层，但仍能产生任何非线性连续函数，以及深度神经网络（DNN），这是一种使用两个以上神经网络层来建模复杂关系的ANN [2]。最后，各种深度学习架构的分类仍然是一个研究问题 [2, 117, 155]

深度学习文献中涵盖的主要架构包括全连接前馈网络（称为FFNN，或也称为FNN、FCNN或FF–DNN）、卷积神经网络（CNNs）和循环神经网络（RNNs） [29, 92]。其他更近期的架构包括自动编码器（AE）、深度信念网络（DBN）、生成对抗网络（GAN）和贝叶斯深度学习（BDL） [5, 17].

基于其中一些网络，已经研究了各种物理信息神经网络（PINN）的扩展。一个示例是使用贝叶斯神经网络估计PINN解的不确定性。或者，在训练卷积神经网络（CNN）和循环神经网络（RNN）时，可以使用有限差分滤波器来构建PINN式损失函数中的微分方程残差。本节将说明所有这些示例以及更多内容；首先，我们将定义表征各种网络的数学框架。

根据Caterini和Chang [25]，一个具有 L 层的通用深度神经网络可以表示为 L 个函数 $f_i(x_i, \theta_i)$，其中 x_i 被称为状态变量，而 θ_i 表示第i层的参数集。因此，一个函数 $u(x)$ 被近似为

$$u_\theta(x) = f_L \circ f_{L-1} \ldots \circ f_1(x). \tag{3}$$

其中每个 f_i 定义在两个内积空间 E_i 和 H_i上，即$f_i \in E_i \times H_i$，并且层组合，由 $\circ$表示，应读作 $f_2 \circ f_1(x) = f_2(f_1(x))$。

由于Raissi等人 [146] 最初的原始基础物理信息神经网络，大多数解决方案都使用了前馈神经网络。然而，一些研究人员尝试了不同类型的神经网络，以查看它们如何影响整体物理信息神经网络的性能，特别是卷积神经网络、循环神经网络和生成对抗网络，并且似乎还没有对物理信息神经网络框架内的其他网络进行充分调查。在本小节中，我们试图综合主流解决方案，利用前馈网络，以及所有其他版本。表 1 包含一个综合报告，报告了神经网络类型及其实现的论文。

2.1.1 前馈神经网络

前馈神经网络，也称为多层感知器（MLP），是一组按层组织的神经元，通过层顺序执行计算。前馈网络，也称为多层感知器（MLP），是具有多个隐藏层的深度神经网络，这些隐藏层仅向前移动（没有回环）。在完全连接的神经网络中，相邻层中的神经元是连接的，而同一层内的神经元则没有连接。每个神经元是一个数学运算，它将自身的加权输入总和加上偏差因子应用于激活函数 [175]。给定一个输入 $x \in \Omega$，MLP将其转换为输出，通过一个由单元（神经元）组成的层，这些单元在连续层之间由仿射线性映射组成，并且在单元内部有标量非线性激活函数，从而形成函数的组合。因此，对于MLP，可以指定（3）为

$$f_i(x_i; W_i, b_i) = \alpha_i(W_i \cdot x_i + b_i)$$

equipping each E_i and H_i with the standard Euclidean inner product, i.e. $E = H = \mathbb{R}$ [26], and α_i is a scalar (non-linear) activation function. The machine learning literature has studied several different activation functions, which we shall discuss later in this section. Equation (3) can also be rewritten in conformity with the notation used in Mishra and Molinaro [111]:

$$u_\theta(x) = C_K \circ \alpha \circ C_{K-1} \ldots \circ \alpha \circ C_1(x), \tag{4}$$

where for any $1 \leq k \leq K$, it it is defined

$$C_k(x_k) = W_k x_k + b_k. \tag{5}$$

Thus a neural network consists of an input layer, an output layer, and $(K - 2)$ hidden layers.

FFNN Architectures

While the ideal DNN architecture is still an ongoing research; papers implementing PINN have attempted to empirically optimise the architecture's characteristics, such as the number of layers and neurons in each layer. Smaller DNNs may be unable to effectively approximate unknown functions, whereas too large DNNs may be difficult to train, particularly with small datasets. Raissi et al [146] used different typologies of DNN, for each problem, like a 5-layer deep neural network with 100 neurons per layer, an DNN with 4 hidden layers and 200 neurons per layer or a 9 layers with 20 neuron each layer. Tartakovsky et al [170] empirically determine the feedforward network size, in particular they use three hidden layers and 50 units per layer, all with an hyperbolic tangent activation function. Another example of how the differential problem affects network architecture can be found in Kharazmi et al [78] for their hp-VPINN. The architecture is implemented with four layers and twenty neurons per layer, but for an advection equation with a double discontinuity of the exact solution, they use an eight-layered deep network. For a constrained approach, by utilizing a specific portion of the NN to satisfy the required boundary conditions, Zhu et al [199] use five hidden layers and 250 neurons per layer to constitute the fully connected neural network. Bringing the number of layers higher, in PINNeik [175], a DNN with ten hidden layers containing twenty neurons each is utilized, with a locally adaptive inverse tangent function as the activation function for all hidden layers except the final layer, which has a linear activation function. He et al [60] examines the effect of neural network size on state estimation accuracy. They begin by experimenting with various hidden layer sizes ranging from three to five, while maintaining a value of 32 neurons per layer. Then they set the number of hidden layers to three, the activation function to hyperbolic tangent, while varying the number of neurons in each hidden layer. Other publications have attempted to understand how the number of layers, neurons, and activation functions effect the NN's approximation quality with respect to the problem to be solved, like [19].

Multiple FFNN

Although many publications employ a single fully connected network, a rising number of research papers have been addressing PINN with multiple fully connected network blended together, e.g. to approximate specific equation of a larger mathematical model. An architecture composed of five the feed-forward neural network is proposed by Haghighat et al [57]. In a two-phase Stefan problem, discussed later in this review and in Cai et al [22], a DNN is used to model the unknown interface between two different material phases, while another DNN describes the two temperature distributions of the phases. Instead of a single NN across

装备每个 E_i 和 H_i 使用标准的欧几里得内积，即 $E = H = \mathbb{R}$ [26]， 和 α_i 是一个标量（非线性）激活函数。机器学习文献已经研究了多种不同的激活函数，我们将在本节后面讨论。方程(3)也可以根据Mishra和Molinaro [111]:中使用的符号重新编写：

$$u_\theta(x) = C_K \circ \alpha \circ C_{K-1} \ldots \circ \alpha \circ C_1(x), \tag{4}$$

其中对于任何 $1 \leq k \leq K$，它被定义为

$$C_k(x_k) = W_k x_k + b_k. \tag{5}$$

因此，一个神经网络由一个输入层、一个输出层和 $(K - 2)$ 个隐藏层组成。

前馈神经网络架构

虽然理想的DNN架构仍然是一个持续的研究课题；但实现PINN的论文已经尝试通过经验优化架构的特性，例如层数和每层的神经元数量。较小的DNN可能无法有效地逼近未知函数，而太大的DNN可能难以训练，尤其是在数据集较小的情况下。Raissi等人 [146] 为每个问题使用了不同类型的DNN，例如每层100个神经元的5层深度神经网络、具有4个隐藏层和每层200个神经元的DNN，或每层20个神经元的9层网络。Tartakovsky等人 [170] 通过经验确定了前馈网络的大小，特别是他们使用了三个隐藏层和每层50个单元，所有隐藏层都采用双曲正切激活函数。Kharazmi等人 [78] 在他们的hp-VPINN中展示了微分问题如何影响网络架构的另一个例子。该架构实现了四层和每层20个神经元，但对于具有精确解双重不连续性的对流方程，他们使用了一个八层的深度网络。对于约束方法，通过利用NN的特定部分来满足所需的边界条件，朱等人 [199] 使用了五个隐藏层和每层250个神经元来构成全连接神经网络。增加层数，在PINNeik [175], 中，一个具有十个隐藏层且每层包含20个神经元DNN被使用，除了最后一层外，所有隐藏层都采用局部自适应反正切函数作为激活函数，最后一层采用线性激活函数。He等人 [60] 研究了神经网络大小对状态估计精度的影响。他们首先通过实验各种隐藏层大小，范围从三到五，同时保持每层32个神经元的值。然后他们设置隐藏层数为三个，激活函数为双曲正切，同时改变每个隐藏层的神经元数量。其他出版物已经尝试理解层数、神经元数量和激活函数如何影响NN的逼近质量，相对于要解决的问题，例如 [19]。

多个FFNN

尽管许多出版物采用单个全连接网络，但越来越多的研究论文正在处理使用多个全连接网络融合的PINN，例如用于逼近更大数学模型的特定方程。哈吉加特等人提出了由五个前馈神经网络组成的架构 [57]。在一个两相斯式藩问题中，正如本综述后续部分和蔡等人 [22], 所述，一个DNN用于模拟两个不同材料相之间的未知界面，而另一个DNN描述了相的两个温度分布。而不是一个单一的NNacross the entire domain, Moseley et al [116] suggests using multiple neural networks, one for each subdomain. Finally, Lu et al [99] employed a pair of DNN, one for encoding the input space (branch net) and the other for encoding the domain of the output functions (trunk net). This architecture, known as DeepONet, is particularly generic because no requirements are made to the topology of the branch or trunk network, despite the fact that the two sub-networks have been implemented as FFNNs as in Lin et al [96].

Shallow Networks

To overcome some difficulties, various researchers have also tried to investigate shallower network solutions: these can be sparse neural networks, instead of fully connected architectures, or more likely single hidden layers as ELM (Extreme Learning Machine) [66]. When compared to the shallow architecture, more hidden layers aid in the modeling of complicated nonlinear relationships [155], however, using PINNs for real problems can result in deep networks with many layers associated with high training costs and efficiency issues. For this reason, not only deep neural networks have been employed for PINNs but also shallow ANN are reported in the literature. X–TFC, developed by Schiassi et al [153], employs a single-layer NN trained using the ELM algorithm. While PIELM [41] is proposed as a faster alternative, using a hybrid neural network-based method that combines two ideas from PINN and ELM. ELM only updates the weights of the outer layer, leaving the weights of the inner layer unchanged.
Finally, in Ramabathiran and Ramachandran [148] a Sparse, Physics-based, and partially Interpretable Neural Networks (SPINN) is proposed. The authors suggest a sparse architecture, using kernel networks, that yields interpretable results while requiring fewer parameters than fully connected solutions. They consider various kernels such as Radial Basis Functions (RBF), softplus hat, or Gaussian kernels, and apply their proof of concept architecture to a variety of mathematical problems.

Activation Function

The activation function has a significant impact on DNN training performance. ReLU, Sigmoid, Tanh are commonly used activations [168]. Recent research has recommended training an adjustable activation function like Swish, which is defined as $x \cdot \text{Sigmoid}(\beta x)$ and β is a trainable parameter, and where Sigmoid is supposed to be a general sigmoid curve, an S-shaped function, or in some cases a logistic function. Among the most used activation functions there are logistic sigmoid, hyperbolic tangent, ReLu, and leaky ReLu. Most authors tend to use the infinitely differentiable hyperbolic tangent activation function $\alpha(x) = \tanh(x)$ [60], whereas Cheng and Zhang [31] use a Resnet block to improve the stability of the fully-connected neural network (FC–NN). They also prove that Swish activation function outperforms the others in terms of enhancing the neural network's convergence rate and accuracy. Nonetheless, because of the second order derivative evaluation, it is pivotal to choose the activation function in a PINN framework with caution. For example, while rescaling the PDE to dimensionless form, it is preferable to choose a range of [0, 1] rather than a wider domain, because most activation functions (such as Sigmoid, Tanh, Swish) are nonlinear near 0. Moreover the regularity of PINNs can be ensured by using smooth activation functions like as the sigmoid and hyperbolic tangent, allowing estimations of PINN generalization error to hold true [111].

整个域，莫斯利等人 [116] 建议使用多个神经网络，每个子域一个。最后，陆等人 [99] 采用了一对深度神经网络，一个用于编码输入空间（分支网络），另一个用于编码输出函数的域（主干网络）。这种架构，称为DeepONet，特别通用，因为它对分支或主干网络的拓扑结构没有要求，尽管这两个子网络在林等人 [96]中已实现为前馈神经网络。

浅层网络

为了克服一些困难，各种研究人员还尝试研究了更浅的网络解决方案：这些可以是稀疏神经网络，而不是全连接架构，或者更有可能是单隐藏层，如极限学习机 (ELM) [66]。与浅层架构相比，更多的隐藏层有助于建模复杂的非线性关系 [155], ，然而，使用物理信息神经网络 (PINNs) 解决实际问题时，可能会产生具有许多层的深层网络，这会导致高训练成本和效率问题。因此，不仅深度神经网络已被用于PINNs，浅层人工神经网络也在文献中有所报道。由Schiassi等人 [153], 开发的X–TFC采用单层神经网络，并使用ELM算法进行训练。虽然PIELM [41] 被提出作为一种更快的替代方案，它使用了一种基于混合神经网络的混合方法，结合了PINN和ELM的两个思想。ELM仅更新外层神经网络的权重，而内层神经网络的权重保持不变。

最后，在拉马巴希兰和拉马钱德拉 [148] 提出了稀疏、基于物理的、部分可解释的神经网络 (SPINN)。作者建议使用核网络的稀疏架构，在需要比全连接解决方案更少参数的同时，得到可解释的结果。他们考虑了各种核，如径向基函数 (RBF)、软帽或高斯核，并将他们的概念验证架构应用于各种数学问题。

激活函数

激活函数对深度神经网络 (DNN) 训练性能有显著影响。ReLU、Sigmoid、Tanh 是常用的激活函数 [168]。最近的研究建议训练一个可调节的激活函数，如 S wish，它被定义为 $x \cdot \text{Sigmoid}(\beta x)$，其中 β 是一个可训练的参数，并且 Sigmoid 应该是一条通用 Sigmoid 曲线、S 形函数，或在某些情况下是一条逻辑函数。在常用的激活函数中，有逻辑 Sigmoid、双曲正切、ReLU 和泄漏 ReLU。大多数作者倾向于使用无限可微的双曲正切激活函数 $\alpha(x) = \tanh(x)$[60], ，而 Cheng 和 Zhang [31] 使用 Resnet 模块来提高全连接神经网络 (FC–NN) 的稳定性。他们还证明，在提高神经网络的收敛率和准确率方面，Swish 激活函数优于其他函数。尽管如此，由于二阶导数评估，在物理信息神经网络 (PINN) 框架中选择激活函数时需要谨慎。例如，在将偏微分方程 (PDE) 转换为无量纲形式时，最好选择一个范围 [0, 1] 而不是更宽的域，因为大多数激活函数（如 Sigmoid、Tanh、Swish）在 0 附近是非线性的。此外，通过使用 sigmoid 和双曲正切等平滑激活函数，可以确保 PINN 的正则性，从而使得对 PINN 泛化误差的估计成立 [111]。

2.1.2 Convolutional Neural Networks

Convolutional neural networks (ConvNet or CNN) are intended to process data in the form of several arrays, for example a color image made of three 2D arrays. CNNs usually have a number of convolution and pooling layers. The convolution layer is made up of a collection of filters (or kernels) that convolve across the full input rather than general matrix multiplication. The pooling layer instead performs subsampling, reducing the dimensionality.

For CNNs, according to Caterini and Chang [26], the layerwise function f can be written as

$$f_i(x_i; W_i) = \Phi_i(\alpha_i(\mathcal{C}_i(W_i, x_i)))$$

where α is an elementwise nonlinearity, Φ is the max-pooling map, and $\mathcal{C}$ the convolution operator.

It is worth noting that the convolution operation preserves translations and pooling is unaffected by minor data translations. This is applied to input image properties, such as corners, edges, and so on, that are translationally invariant, and will still be represented in the convolution layer's output.

As a result, CNNs perform well with multidimensional data such as images and speech signals, in fact is in the domain of images that these networks have been used in a physic informed network framework.

For more on these topic the reader can look at LeCun et al [92], Chen et al [29], Muhammad et al [117], Aldweesh et al [2], Berman et al [17], Calin [23]

CNN Architectures

Because CNNs were originally created for image recognition, they are better suited to handling image-like data and may not be directly applicable to scientific computing problems, as most geometries are irregular with non-uniform grids; for example, Euclidean-distance-based convolution filters lose their invariance on non-uniform meshes.

A physics-informed geometry-adaptive convolutional neural network (PhyGeoNet) was introduced in Gao et al [48]. PhyGeoNet is a physics-informed CNN that uses coordinate transformation to convert solution fields from an irregular physical domain to a rectangular reference domain. Additionally, boundary conditions are strictly enforced making it a physics-constrained neural network.

Fang [44] observes that a Laplacian operator has been discretized in a convolution operation employing a finite-difference stencil kernel. Indeed, a Laplacian operator can be discretized using the finite volume approach, and the discretization procedure is equivalent to convolution. As a result, he enhances PINN by using a finite volume numerical approach with a CNN structure. He devised and implemented a CNN-inspired technique in which, rather than using a Cartesian grid, he computes convolution on a mesh or graph. Furthermore, rather than zero padding, the padding data serve as the boundary condition. Finally, Fang [44] does not use pooling because the data does not require compression.

Convolutional Encoder–Decoder Network

Autoencoders (AE) (or encoder–decoders) are commonly used to reduce dimensionality in a nonlinear way. It consists of two NN components: an encoder that translates the data from

2.1.2 卷积神经网络

卷积神经网络（ConvNet或CNN）旨在处理以多个数组形式的数据，例如由三个2D数组组成的彩色图像。CNNs通常包含多个卷积层和池化层。卷积层由一组滤波器（或核）组成，这些滤波器在整个输入上进行卷积，而不是一般的矩阵乘法。池化层则执行下采样，降低维度。

对于CNNs，根据Caterini和Chang [26], ，逐层的函数 f 可以表示为

$$f_i(x_i; W_i) = \Phi_i(\alpha_i(\mathcal{C}_i(W_i, x_i)))$$

其中 α 是逐元素非线性， Φ 是最大池化映射，而 $\mathcal{C}$ 是卷积算子。

值得注意的是，卷积操作保持平移，池化操作不受微小数据平移的影响。这适用于输入图像属性，如角点、边缘等，这些属性在平移下是不变的，并且仍然会在卷积层的输出中体现。

因此，卷积神经网络在图像和语音信号等多维数据上表现良好，实际上这些网络在物理信息网络框架中已被应用于图像领域。

关于这些主题的更多信息，读者可以参考LeCun等人 [92], 陈等人 [29], 穆罕默德等人 [117], 阿尔德维什等人 [2], 伯曼等人 [17], 卡林 [23]

CNN架构

因为卷积神经网络最初是为图像识别而创建的，它们更适合处理图像类数据，可能不直接适用于科学计算问题，因为大多数几何形状是不规则的，具有非均匀网格；例如，基于欧几里得距离的卷积滤波器在非均匀网格上会失去其不变性。

在 Gao 等人 [48] 中介绍了一种物理信息几何自适应卷积神经网络（PhyGeoNet）。PhyGeoNet 是一种物理信息卷积神经网络，它使用坐标变换将解场从不规则物理域转换为矩形参考域。此外，边界条件被严格执行，使其成为物理约束神经网络。

方 [44] 观察到在采用有限差分模板核的卷积操作中，拉普拉斯算子已被离散化。实际上，拉普拉斯算子可以使用有限体积方法进行离散化，离散化过程等效于卷积。因此，他通过使用具有卷积神经网络结构的有限体积数值方法增强了 PI NN。他设计并实现了一种受卷积神经网络启发的技术，其中，他不是使用笛卡尔网格，而是在网格或图上计算卷积。此外，他不是使用零填充，而是将填充数据作为边界条件。最后，方 [44] 不使用池化，因为数据不需要压缩。

卷积编码器-解码器网络

自动编码器（AE）（或编码器-解码器）通常用于非线性地降低维度。它由两个神经网络组件组成：一个编码器，该编码器将数据从

the input layer to a finite number of hidden units, and a decoder that has an output layer with the same number of nodes as the input layer [29].

For modeling stochastic fluid flows, Zhu et al [200] developed a physics-constrained convolutional encoder–decoder network and a generative model. Zhu et al [200] propose a CNN-based technique for solving stochastic PDEs with high-dimensional spatially changing coefficients, demonstrating that it outperforms FC–NN methods in terms of processing efficiency.
AE architecture of the convolutional neural network (CNN) is also used in Wang et al [177]. The authors propose a framework called a Theory-guided Auto–Encoder (TgAE) capable of incorporating physical constraints into the convolutional neural network.

Geneva and Zabaras [51] propose a deep auto-regressive dense encoder–decoder and the physics-constrained training algorithm for predicting transient PDEs. They extend this model to a Bayesian framework to quantify both epistemic and aleatoric uncertainty. Finally, Grubišić et al [54] also used an encoder–decoder fully convolutional neural network.

2.1.3 Recurrent Neural Networks

Recurrent neural network (RNN) is a further ANN type, where unlike feed-forward NNs, neurons in the same hidden layer are connected to form a directed cycle. RNNs may accept sequential data as input, making them ideal for time-dependent tasks [29]. The RNN processes inputs one at a time, using the hidden unit output as extra input for the next element [17]. An RNN's hidden units can keep a state vector that holds a memory of previous occurrences in the sequence.

It is feasible to think of RNN in two ways: first, as a state system with the property that each state, except the first, yields an outcome; secondly, as a sequence of vanilla feedforward neural networks, each of which feeds information from one hidden layer to the next. For RNNs, according to Caterini and Chang [26], the layerwise function f can be written as

$$f_i(h_{i-1}) = \alpha(W \cdot h_{i-1} + U \cdot x_i + b).$$

where α is an elementwise nonlinearity (a typical choice for RNN is the tanh function), and where the hidden vector state h evolves according to a hidden-to-hidden weight matrix W, which starts from an input-to-hidden weight matrix U and a bias vector b.

RNNs have also been enhanced with several memory unit types, such as long short time memory (LSTM) and gated recurrent unit (GRU) [2]. Long short-term memory (LSTM) units have been created to allow RNNs to handle challenges requiring long-term memories, since LSTM units have a structure called a memory cell that stores information.
Each LSTM layer has a set of interacting units, or cells, similar to those found in a neural network. An LSTM is made up of four interacting units: an internal cell, an input gate, a forget gate, and an output gate. The cell state, controlled by the gates, can selectively propagate relevant information throughout the temporal sequence to capture the long short-term time dependency in a dynamical system [197].

The gated recurrent unit (GRU) is another RNN unit developed for extended memory; GRUs are comparable to LSTM, however they contain fewer parameters and are hence easier to train [17].

RNN Architectures

Viana et al [174] introduce, in the form of a neural network, a model discrepancy term into a given ordinary differential equation. Recurrent neural networks are seen as ideal for

将输入层映射到有限数量的隐藏单元，并解码器具有与输入层相同节点数的输出层 [29]。

为了对随机流体流动进行建模，朱等人 [200] 开发了一个物理约束卷积编码器-解码器网络和一个生成模型。朱等人 [200] 提出了一种基于卷积神经网络(CNN)的技术，用于求解具有高维空间变化系数的随机偏微分方程，并证明它在处理效率方面优于全连接-神经网络方法。

卷积神经网络(CNN)的自动编码器架构也用于王等人 [177]。作者提出了一种称为理论指导自动编码器(TgAE)的框架，能够将物理约束集成到卷积神经网络中。

日内瓦和扎巴拉斯 [51] 提出了一种深度自回归密集编码器-解码器以及物理约束训练算法，用于预测瞬态偏微分方程。他们将此模型扩展到贝叶斯框架，以量化认知不确定性和偶然不确定性。最后，Grubiši´c 等人 [54] 也使用了一个编码器-解码器全卷积神经网络。

2.1.3 循环神经网络

循环神经网络(RNN)是一种进一步的人工神经网络类型，与前馈神经网络不同，同一隐藏层中的神经元会形成有向循环。循环神经网络可以接受序列数据作为输入，使其非常适合时间依赖性任务 [29]。循环神经网络逐个处理输入，使用隐藏单元输出作为下一个元素的额外输入 [17]。循环神经网络的隐藏单元可以保持一个状态向量，该向量保存了序列中先前出现的信息。

可以以两种方式理解循环神经网络：首先，作为一个状态系统，其中每个状态（除第一个外）都会产生一个结果；其次，作为一系列香草前馈神经网络，每个网络将一个隐藏层的信息传递到下一个隐藏层。对于循环神经网络，根据 Caterini 和 Chang [26], ，层状函数 f 可以写成

$$f_i(h_{i-1}) = \alpha(W \cdot h_{i-1} + U \cdot x_i + b).$$

其中 α 是一个逐元素非线性（循环神经网络通常选择tanh函数），并且隐藏向量状态 h 根据隐藏到隐藏的权重矩阵 W演变，该矩阵从输入到隐藏的权重矩阵 U 和偏置向量 b开始。

循环神经网络（RNNs）也通过多种记忆单元类型得到了增强，例如长短期记忆（LSTM）和门控循环单元（GRU） [2]。长短期记忆（LSTM）单元被创建出来，以使RNNs能够处理需要长期记忆的挑战，因为LSTM单元具有一个称为记忆单元的结构，该结构用于存储信息。

每个LSTM层有一组交互单元，或称为单元，类似于神经网络中发现的单元。一个LSTM由四个交互单元组成：一个内部单元、一个输入门、一个遗忘门和一个输出门。单元状态由门控制，可以选择性地在整个时间序列中传播相关信息，以捕获动力系统中的长短期时间依赖性 [197]。

门控循环单元（GRU）是另一种为扩展记忆而开发的RNN单元；GRUs与LSTM相当，但它们包含更少的参数，因此更容易训练 [17]。

RNN架构

维亚纳等人 [174] 以神经网络的形式，将一个模型差异项引入给定的常微分方程。循环神经网络被认为是理想的

dynamical systems because they expand classic feedforward networks to incorporate time-dependent responses. Because a recurrent neural network applies transformations to given states in a sequence on a periodic basis, it is possible to design a recurrent neural network cell that does Euler integration; in fact, physics-informed recurrent neural networks can be used to perform numerical integration.

Viana et al [174] build recurrent neural network cells in such a way that specific numerical integration methods (e.g., Euler, Riemann, Runge–Kutta, etc.) are employed. The recurrent neural network is then represented as a directed graph, with nodes representing individual kernels of the physics-informed model. The graph created for the physics-informed model can be used to add data-driven nodes (such as multilayer perceptrons) to adjust the outputs of certain nodes in the graph, minimizing model discrepancy.

LSTM Architectures

Physicists have typically employed distinct LSTM networks to depict the sequence-to-sequence input-output relationship; however, these networks are not homogeneous and cannot be directly connected to one another. In Zhang et al [197] this relationship is expressed using a single network and a central finite difference filter-based numerical differentiator. Zhang et al [197] show two architectures for representation learning of sequence-to-sequence features from limited data that is augmented by physics models. The proposed networks is made up of two ($PhyLSTM^2$) or three ($PhyLSTM^3$) deep LSTM networks that describe state space variables, nonlinear restoring force, and hysteretic parameter. Finally, a tensor differentiator, which determines the derivative of state space variables, connects the LSTM networks.

Another approach is Yucesan and Viana [194] for temporal integration, that implement an LSTM using a previously introduced Euler integration cell.

2.1.4 Other Architectures for PINN

Apart from fully connected feed forward neural networks, convolutional neural networks, and recurrent neural networks, this section discusses other approaches that have been investigated. While there have been numerous other networks proposed in the literature, we discovered that only Bayesian neural networks (BNNs) and generative adversarial networks (GANs) have been applied to PINNs. Finally, an interesting application is to combine multiple PINNs, each with its own neural network.

Bayesian Neural Network

In the Bayesian framework, Yang et al [190] propose to use Bayesian neural networks (BNNs), in their B-PINNs, that consists of a Bayesian neural network subject to the PDE constraint that acts as a prior. BNN are neural networks with weights that are distributions rather than deterministic values, and these distributions are learned using Bayesian inference. For estimating the posterior distributions, the B-PINN authors use the Hamiltonian Monte Carlo (HMC) method and the variational inference (VI). Yang et al [190] find that for the posterior estimate of B-PINNs, HMC is more appropriate than VI with mean field Gaussian approximation.

They analyse also the possibility to use the Karhunen–Loève expansion as a stochastic process representation, instead of BNN. Although the KL is as accurate as BNN and considerably quicker, it cannot be easily applied to high-dimensional situations. Finally, they

动力系统，因为它们扩展了经典前馈网络以包含时变响应。由于循环神经网络以周期性为基础对序列中的给定状态应用转换，因此可以设计一个执行欧拉积分的循环神经网络单元；事实上，物理信息循环神经网络可用于执行数值积分。

维亚纳等人 [174] 以某种方式构建循环神经网络单元，以便采用特定的数值积分方法（例如，欧拉、黎曼、龙格-库塔等）。然后，循环神经网络表示为一个有向图，节点代表物理信息模型的单个核。为物理信息模型创建的图可用于添加数据驱动节点（如多层感知器），以调整图中某些节点的输出，从而最小化模型差异。

LSTM架构

物理学家通常采用不同的LSTM网络来描述序列到序列的输入输出关系；然而，这些网络不是同质的，不能直接连接。在张等人 [197] 中，这种关系使用单个网络和一个基于有限差分滤波器的数值微分器表示。张等人 [197] 展示了两种用于从受物理模型增强的有限数据中学习序列到序列特征的架构。所提出的网络由两个（$PhyLSTM$2）或三个（$PhyLSTM$3）描述状态空间变量、非线性恢复力和迟滞参数的深度LSTM网络组成。最后，一个张量微分器，它确定状态空间变量的导数，连接了LSTM网络。

另一种方法是余和Viana [194] 用于时间积分，该方法是使用先前介绍过的欧拉积分单元来实现LSTM。

2.1.4 PINN的其他架构

除了全连接前馈神经网络、卷积神经网络和循环神经网络之外，本节讨论了其他已被研究的方法。虽然文献中提出了许多其他网络，但我们发现只有贝叶斯神经网络（BNNs）和生成对抗网络（GANs）被应用于PINNs。最后，一个有趣的应用是将多个PINNs结合，每个PINN都有自己的神经网络。

贝叶斯神经网络

在贝叶斯框架中，杨等人 [190] 提出在他们的B-PINNs中使用贝叶斯神经网络（BNNs），该网络由一个受PDE约束的贝叶斯神经网络组成，该约束作为先验。BNN是权重为分布而不是确定性值的神经网络，这些分布是使用贝叶斯推理学习的。为了估计后验分布，B-PINN作者使用了哈密顿蒙特卡洛（HMC）方法和变分推理（VI）。杨等人 [190] 发现，对于B-PINNs的后验估计，HMC比具有均值场高斯近似的VI更合适。

他们还分析了使用Karhunen-Loève展开作为随机过程表示的可能性，而不是BNN。尽管KL和BNN一样不准确，但速度要快得多，但它不能轻易地应用于高维情况。最后，他们

observe that to estimate the posterior of a Bayesian framework, KL–HMC or deep normalizing flow (DNF) models can be employed. While DNF is more computationally expensive than HMC, it is more capable of extracting independent samples from the target distribution after training. This might be useful for data-driven PDE solutions, however it is only applicable to low-dimensional problems.

GAN Architectures

In generative adversarial networks (GANs), two neural networks compete in a zero-sum game to deceive each other. One network generates and the other discriminates. The generator accepts input data and outputs data with realistic characteristics. The discriminator compares the real input data to the output of the generator. After training, the generator can generate new data that is indistinguishable from real data [17].

Yang et al [189] propose a new class of generative adversarial networks (PI–GANs) to address forward, inverse, and mixed stochastic problems in a unified manner. Unlike typical GANs, which rely purely on data for training, PI–GANs use automatic differentiation to embed the governing physical laws in the form of stochastic differential equations (SDEs) into the architecture of PINNs. The discriminator in PI–GAN is represented by a basic FFNN, while the generators are a combination of FFNNs and a NN induced by the SDE.

Multiple PINNs

A final possibility is to combine several PINNs, each of which could be implemented using a different neural network. Jagtap et al [71] propose a conservative physics-informed neural network (cPINN) on discrete domains. In this framework, the complete solution is recreated by patching together all of the solutions in each sub-domain using the appropriate interface conditions. This type of domain segmentation also allows for easy network parallelization, which is critical for obtaining computing efficiency. This method may be expanded to a more general situation, called by the authors as Mortar PINN, for connecting non-overlapping deconstructed domains. Moreover, the suggested technique may use totally distinct neural networks, in each subdomain, with various architectures to solve the same underlying PDE. Stiller et al [167] proposes the GatedPINN architecture by incorporating conditional computation into PINN. This architecture design is composed of a gating network and set of PINNs, hereinafter referred to as "experts"; each expert solves the problem for each space-time point, and their results are integrated via a gating network. The gating network determines which expert should be used and how to combine them. We will use one of the expert networks as an example in the following section of this review.

2.2 Injection of Physical Laws

To solve a PDE with PINNs, derivatives of the network's output with respect to the inputs are needed. Since the function u is approximated by a NN with smooth activation function, $\hat{u}_\theta$, it can be differentiated. There are four methods for calculating derivatives: hand-coded, symbolic, numerical, and automatic. Manually calculating derivatives may be correct, but it is not automated and thus impractical [175].

Symbolic and numerical methods like finite differentiation perform very badly when applied to complex functions; automatic differentiation (AD), on the other hand, overcomes numerous restrictions as floating-point precision errors, for numerical differentiation, or

观察到，为了估计贝叶斯框架的后验，可以使用KL-HMC或深度归一化流（DNF）模型。虽然DNF比HMC计算成本更高，但在训练后，它更能够从目标分布中提取独立样本。这对于数据驱动的偏微分方程解可能很有用，但它仅适用于低维问题。

GAN 架构

在生成对抗网络（GANs）中，两个神经网络在一个零和博弈中相互欺骗。一个网络生成数据，另一个网络进行判别。生成器接受输入数据并输出具有真实特征的数据。判别器将真实输入数据与生成器的输出进行比较。训练后，生成器可以生成与真实数据无法区分的新数据 [17]。

杨等人 [189] 提出了一种新的生成对抗网络（PI–GANs）类别，以统一方式解决正向、逆向和混合随机问题。与依赖纯数据进行训练的典型GANs不同，PI–GANs使用自动微分将控制物理定律以随机微分方程（SDEs）的形式嵌入到PINNs的架构中。PI–GAN中的判别器由一个基本的FFNN表示，而生成器是FFNNs和由SDE诱导的NN的组合。

多个PINN

一种最终的可能性是将多个PINN组合起来，每个PINN都可以使用不同的神经网络来实现。Jagtap等人 [71] 在离散域上提出了一种保守的物理信息神经网络（cPINN）。在这个框架中，通过使用适当的界面条件将每个子域中的所有解拼接在一起来重新创建完整解。这种域分割还允许轻松的网络并行化，这对于获得计算效率至关重要。这种方法可以扩展到作者称为锚钉PINN（Mortar PINN）的更一般的情况，用于连接非重叠的解构域。此外，所建议的技术可以在每个子域中使用完全不同的神经网络，具有不同的架构来解决相同的底层偏微分方程（PDE）。Stiller等人 [167] 通过将条件计算纳入PINN中提出了门控PINN（GatedPINN）架构。这种架构设计由一个门控网络和一组PINN组成，此处统称为"专家"；每个专家在每个时空点解决该问题，并通过门控网络集成其结果。门控网络确定应使用哪个专家以及如何组合它们。在本综述的下一节中，我们将使用其中一个专家网络作为示例。

2.2 物理定律的注入

要使用物理信息神经网络（PINN）求解偏微分方程（PDE），需要计算网络输出相对于输入的导数。由于函数 u 由具有平滑激活函数的神经网络（NN）$\hat{}u_\theta$ 近似，因此可以对其进行微分。计算导数有四种方法：手写、符号、数值和自动。手动计算导数可能正确，但它不是自动化的，因此不实用 [175]。

符号和数值方法（如有限差分法）在应用于复杂函数时表现非常差；而自动微分（AD）则克服了数值微分中的许多限制，如浮点精度误差，或

Table 1 The main neural network utilized in PINN implementations is synthesized in this table

NN family	NN type	Papers
FF–NN	1 layer / EML	Dwivedi and Srinivasan [41], Schiassi et al [153]
	2–4 layers	32 neurons per layer He et al [60] 50 neurons per layer Tartakovsky et al [170]
	5–8 layers	250 neurons per layer Zhu et al [199]
	9+ layers	Cheng and Zhang [31] Waheed et al [175]
	Sparse	Ramabathiran and Ramachandran [148]
	multi FC-DNN	Amini Niaki et al [6] Islam et al [69]
CNN	plain CNN	Gao et al [48] Fang [44]
	AE CNN	Zhu et al [200], Geneva and Zabaras [51] Wang et al [177]
RNN	RNN	Viana et al [174]
	LSTM	Zhang et al [197] Yucesan and Viana [194]
Other	BNN	Yang et al [190]
	GAN	Yang et al [189]

We summarize Sect. 2 by showcasing some of the papers that represent each of the many Neural Network implementations of PINN. Feedforward neural networks (FFNNs), convolutional neural networks (CNNs), and recurrent neural networks (RNN) are the three major families of Neural Networks reported here. A publication is reported for each type that either used this type of network first or best describes its implementation. In the literature, PINNs have mostly been implemented with FFNNs with 5–10 layers. CNN appears to have been applied in a PCNN manner for the first time, by incorporating the boundary condition into the neural network structure rather than the loss

Table 1 The main neural network utilized in PINN implementations is synthesized in this table

NN family	NN type	Papers
FF–NN	1 layer / EML	Dwivedi and Srinivasan [41], Schiassi et al [153]
	2–4 layers	32 neurons per layer He et al [60] 50 neurons per layer Tartakovsky et al [170]
	5–8 layers	250 neurons per layer Zhu et al [199]
	9+ layers	Cheng and Zhang [31] Waheed et al [175]
	Sparse	Ramabathiran and Ramachandran [148]
	multi FC-DNN	Amini Niaki et al [6] Islam et al [69]
CNN	plain CNN	Gao et al [48] Fang [44]
	AE CNN	Zhu et al [200], Geneva and Zabaras [51] Wang et al [177]
RNN	RNN	Viana et al [174]
	LSTM	Zhang et al [197] Yucesan and Viana [194]
Other	BNN	Yang et al [190]
	GAN	Yang et al [189]

We summarize Sect. 2 by showcasing some of the papers that represent each of the many Neural Network implementations of PINN. Feedforward neural networks (FFNNs), convolutional neural networks (CNNs), and recurrent neural networks (RNN) are the three major families of Neural Networks reported here. A publication is reported for each type that either used this type of network first or best describes its implementation. In the literature, PINNs have mostly been implemented with FFNNs with 5–10 layers. CNN appears to have been applied in a PCNN manner for the first time, by incorporating the boundary condition into the neural network structure rather than the loss

memory intensive symbolic approach. AD use exact expressions with floating-point values rather than symbolic strings, there is no approximation error [175].

Automatic differentiation (AD) is also known as autodiff, or algorithmic differentiation, although it would be better to call it algorithmic differentiation since AD does not totally automate differentiation: instead of symbolically evaluating derivatives operations, AD performs an analytical evaluation of them.

Considering a function $f : \mathbb{R}^n \to \mathbb{R}^m$ of which we want to calculate the Jacobian J, after calculating the graph of all the operations composing the mathematical expression, AD can then work in either forward or reverse mode for calculating the numerical derivative.

AD results being the main technique used in literature and used by all PINN implementations, in particular only Fang [44] use local fitting method approximation of the differential operator to solve the PDEs instead of automatic differentiation (AD). Moreover, by using local fitting method rather than employing automated differentiation, Fang is able to verify that a his PINN implementation has a convergence.

Essentially, AD incorporates a PDE into the neural network's loss Eq. (2), where the differential equation residual is

$$r_{\mathcal{F}}[\hat{u}_\theta](z) = r_\theta(z) := \mathcal{F}(\hat{u}_\theta(z); \gamma) - f,$$

and similarly the residual NN corresponding to boundary conditions (BC) or initial conditions (IC) is obtained by substituting $\hat{u}_\theta$ in the second equation of (1), i.e.

$$r_{\mathcal{B}}[\hat{u}_\theta](z) := \mathcal{B}(\hat{u}_\theta(z)) - g(z).$$

Using these residuals, it is possible to assess how well an approximation u_θ satisfies (1). It is worth noting that for the exact solution, u, the residuals are $r_{\mathcal{F}}[u] = r_{\mathcal{B}}[u] = 0$ [38].

In Raissi and Karniadakis [140], Raissi et al [146], the original formulation of the aforementioned differential equation residual, leads to the form of

$$r_{\mathcal{F}}[\hat{u}_\theta](z) = r_\theta(x,t) = \frac{\partial}{\partial t}\hat{u}_\theta(x,t) + \mathcal{F}_x \hat{u}_\theta(x,t).$$

In the deep learning framework, the principle of imposing physical constraints is represented by differentiating neural networks with respect to input spatiotemporal coordinates using the chain rule. In Mathews et al [106] the model loss functions are embedded and then further normalized into dimensionless form. The repeated differentiation, with AD, and composition of networks used to create each individual term in the partial differential equations results in a much larger resultant computational graph. As a result, the cumulative computation graph is effectively an approximation of the PDE equations [106].

The chain rule is used in automatic differentiation for several layers to compute derivatives hierarchically from the output layer to the input layer. Nonetheless, there are some situations in which the basic chain rule does not apply. Pang et al [128] substitute fractional differential operators with their discrete versions, which are subsequently incorporated into the PINNs' loss function.

2.3 Model Estimation by Learning Approaches

The PINN methodology determines the parameters θ of the NN, $\hat{u}_\theta$, by minimizing a loss function, i.e.

$$\theta = \arg\min_\theta \mathcal{L}(\theta)$$

内存密集型符号方法。自动微分（AD）使用浮点值精确表达式而不是符号字符串，因此没有近似误差 [175]。

自动微分（AD）也称为自动微分，或算法微分，但最好称为算法微分，因为AD并不完全自动微分：它不是通过符号计算导数运算，而是对它们进行解析评估。

考虑一个函数 $f : \mathbb{R}^n \to \mathbb{R}^m$，我们想计算其雅可比矩阵 J，在计算组成数学表达式的所有运算的图之后，AD可以在正向或反向模式下工作以计算数值导数。

AD结果是文献中使用的主要技术，也是所有物理信息神经网络（PINN）实现使用的技术，特别是只有方 [44] 使用局部拟合方法近似微分算子来求解偏微分方程（PDEs），而不是自动微分（AD）。此外，通过使用局部拟合方法而不是采用自动微分，方能够验证他的PINN实现具有收敛性。

本质上，AD 将偏微分方程（PDE）纳入神经网络的损失函数 Eq. (2) 中，其中微分方程残差是

$$r_{\mathcal{F}}[\hat{u}_\theta](z) = r_\theta(z) := \mathcal{F}(\hat{u}_\theta(z); \gamma) - f,$$

类似地，对应边界条件（BC）或初始条件（IC）的残差NN是通过将 $\hat{}u_\theta$ 代入（1）的第二方程中获得的，即

$$r_{\mathcal{B}}[\hat{u}_\theta](z) := \mathcal{B}(\hat{u}_\theta(z)) - g(z).$$

利用这些残差，可以评估近似u_θ 是否满足（1）。值得注意的是，对于精确解u，残差是 $r_{\mathcal{F}}[u] = r_{\mathcal{B}}[u] = 0$ [38]。

在 Raissi 和 Karniadakis [140], Raissi 等人 [146], 中，上述微分方程残差的原始公式导致的形式是

$$r_{\mathcal{F}}[\hat{u}_\theta](z) = r_\theta(x,t) = \frac{\partial}{\partial t}\hat{u}_\theta(x,t) + \mathcal{F}_x \hat{u}_\theta(x,t).$$

在深度学习框架中，施加物理约束的原则是通过链式法则对神经网络的输入时空坐标进行微分来表示。在 Mathews 等人 [106] 中，模型损失函数被嵌入，然后进一步归一化为无量纲形式。使用 AD 进行重复微分，并组合用于创建偏微分方程中的每个单独项的网络，导致结果计算图大大增大。因此，累积计算图有效地是偏微分方程 [106] 的近似。

链式法则在自动微分中用于多层，以从输出层到输入层逐层计算导数。然而，在某些情况下，基本的链式法则并不适用。Pang等人 [128] 用它们的离散版本替换分数阶微分算子，这些离散版本随后被纳入PINNs的损失函数中。

2.3 基于学习方法的模型估计

PINN方法通过最小化损失函数来确定神经网络的参数 θ， $\hat{}u_\theta$，即

$$\theta = \arg\min_\theta \mathcal{L}(\theta)$$

where

$$\mathcal{L}(\theta) = \omega_{\mathcal{F}}\mathcal{L}_{\mathcal{F}}(\theta) + \omega_{\mathcal{B}}\mathcal{L}_{\mathcal{B}}(\theta) + \omega_d\mathcal{L}_{data}(\theta). \tag{6}$$

The three terms of $\mathcal{L}$ refer to the errors in describing the initial $\mathcal{L}_i$ or boundary condition $\mathcal{L}_b$, both indicated as $\mathcal{L}_{\mathcal{B}}$, the loss respect the partial differential equation $\mathcal{L}_{\mathcal{F}}$, and the validation of known data points $\mathcal{L}_{data}$. Losses are usually defined in the literature as a sum, similar to the previous equations, however they can be considered as integrals

$$\mathcal{L}_{\mathcal{F}}(\theta) = \int_{\bar{\Omega}} \left(\mathcal{F}(\hat{u}_\theta(z)) - f(z_i)\right)^2 dz$$

This formulation is not only useful for a theoretical study, as we will see in 2.4, but it is also implemented in a PINN package, NVIDIA Modulus [123], allowing for more effective integration strategies, such as sampling with higher frequency in specific areas of the domain to more efficiently approximate the integral losses.

Note that, if the PINN framework is employed as a supervised methodology, the neural network parameters are chosen by minimizing the difference between the observed outputs and the model's predictions; otherwise, just the PDE residuals are taken into account.

As in Eq. (6) the first term, $\mathcal{L}_{\mathcal{F}}$, represents the loss produced by a mismatch with the governing differential equations $\mathcal{F}$ [60, 167]. It enforces the differential equation $\mathcal{F}$ at the *collocation points*, which can be chosen uniformly or unevenly over the domain Ω of Eq. (1).

The remaining two losses attempt to fit the known data over the NN. The loss caused by a mismatch with the data (i.e., the measurements of u) is denoted by $\mathcal{L}_{data}(\theta)$. The second term typically forces $\hat{u}_\theta$ to mach the measurements of u over provided points (z, u^*), which can be given as synthetic data or actual measurements, and the weight ω_d can account for the quality of such measurements.

The other term is the loss due to mismatch with the boundary or initial conditions, $\mathcal{B}(\hat{u}_\theta) = g$ from Eq. (1).

Essentially, the training approach recovers the shared network parameters θ from:

- few scattered observations of $u(z)$, specifically $\{z_i, u_i^*\}$, $i = 1, \dots, N_d$
- as well as a greater number of collocation points $\{z_i, r_i = 0\}$, $i = 1, \dots, N_r$ for the residual,

The resulting optimization problem can be handled using normal stochastic gradient descent without the need for constrained optimization approaches by minimizing the combined loss function. A typical implementation of the loss uses a mean square error formulation [82], where:

$$\mathcal{L}_{\mathcal{F}}(\theta) = MSE_{\mathcal{F}} = \frac{1}{N_c}\sum_{i=1}^{N_c}\|\mathcal{F}(\hat{u}_\theta(z_i)) - f(z_i)\|^2 = \frac{1}{N_c}\sum_{i=1}^{N_c}\|r_\theta(u_i) - r_i\|^2$$

enforces the PDE on a wide set of randomly selected collocation locations inside the domain, i.e. penalizes the difference between the estimated left-hand side of a PDE and the known right-hand side of a PDE [82]; other approaches may employ an integral definition of the loss [61]. As for the boundary and initial conditions, instead

$$\mathcal{L}_{\mathcal{B}}(\theta) = MSE_{\mathcal{B}} = \frac{1}{N_B}\sum_{i=1}^{N_B}\|\mathcal{B}(\hat{u}_\theta(z)) - g(z_i)\|^2$$

在

$$\mathcal{L}(\theta) = \omega_{\mathcal{F}}\mathcal{L}_{\mathcal{F}}(\theta) + \omega_{\mathcal{B}}\mathcal{L}_{\mathcal{B}}(\theta) + \omega_d\mathcal{L}_{data}(\theta). \tag{6}$$

公式$\mathcal{L}$ 中的三个项指的是描述初始条件 $\mathcal{L}_i$ 或边界条件 $\mathcal{L}_b$时的误差，均表示为$\mathcal{L}_{\mathcal{B}}$，即对偏微分方程 $\mathcal{L}_{\mathcal{F}}$的损失，以及已知数据点的验证 $\mathcal{L}$数据。文献中通常将损失定义为类似于先前公式的总和，但也可以将其视为积分

$$\mathcal{L}_{\mathcal{F}}(\theta) = \int_{\bar{\Omega}} \left(\mathcal{F}(\hat{u}_\theta(z)) - f(z_i)\right)^2 dz$$

这种公式不仅适用于理论研究，正如我们将在2.4节中看到的那样，它还实现于一个物理信息神经网络包中，即NVIDIA Modulus [123], ，允许采用更有效的积分策略，例如在域的特定区域进行更高频率的采样，以更有效地逼近积分损失。

请注意，如果将物理信息神经网络框架作为监督方法使用，则通过最小化观测输出与模型预测之间的差异来选择神经网络参数；否则，仅考虑偏微分方程残差。

如公式（6）所示，第一项，$\mathcal{L}_{\mathcal{F}}$，表示与控制微分方程不匹配所产生的损失$\mathcal{F}$ [60, 167]。它强制在配置点 $\mathcal{F}$ 处满足微分方程配置点，这些点可以在公式（<style id='20'>1<style id='22'>）的域 Ω 上均匀或不均匀地选择。

其余两个损失试图在神经网络上拟合已知数据。与数据（即u的测量值）不匹配所引起的损失用 $\mathcal{L}_{data}(\theta)$表示。第二项通常迫使 $\hat{u}_\theta$ 在提供的点 (z,u^*)上与测量值匹配，这些点可以作为合成数据或实际测量值，权重 ω_d 可以反映这些测量的质量。

另一项是与边界或初始条件不匹配所产生的损失，$\mathcal{B}(\hat{u}_\theta) = g$ 来自公式（1）。

本质上，训练方法从以下内容中恢复共享网络参数 θ：

● 少量分散的观测数据 $u(z)$，具体为 $\{z_i, u^*_i\}$，$i = 1, \dots, N_d$ ● 以及更多的配置点 $\{z_i, r_i = 0\}$，$i = 1, \dots, N_r$用于残差，

由此产生的优化问题可以使用标准的随机梯度下降来处理，而无需使用约束优化方法，通过最小化组合损失函数。损失的一个典型实现使用均方误差公式[82], ，其中：

$$\mathcal{L}_{\mathcal{F}}(\theta) = MSE_{\mathcal{F}} = \frac{1}{N_c}\sum_{i=1}^{N_c}\|\mathcal{F}(\hat{u}_\theta(z_i)) - f(z_i)\|^2 = \frac{1}{N_c}\sum_{i=1}^{N_c}\|r_\theta(u_i) - r_i\|^2$$

在域内的一组随机选择的配置位置上强制执行偏微分方程，即惩罚偏微分方程估计的左侧与已知右侧之间的差异 [82]；其他方法可能采用积分定义的损失 [61]。至于边界和初始条件，相反

$$\mathcal{L}_{\mathcal{B}}(\theta) = MSE_{\mathcal{B}} = \frac{1}{N_B}\sum_{i=1}^{N_B}\|\mathcal{B}(\hat{u}_\theta(z)) - g(z_i)\|^2$$

whereas for the data points,

$$\mathcal{L}_{data}(\theta) = MSE_{data} = \frac{1}{N_d}\sum_{i=1}^{N_d} \|\hat{u}_\theta(z_i) - u_i^*\|^2.$$

computes the error of the approximation $u(t, x)$ at known data points. In the case of a forward problem, the data loss might also indicate the boundary and initial conditions, while in an inverse problem it refers to the solution at various places inside the domain [82].

In Raissi and Karniadakis [140], Raissi et al [146], original approach the overall loss (6) was formulated as

$$\mathcal{L}(\theta) = \frac{1}{N_c}\sum_{i=1}^{N_c} \|\frac{\partial}{\partial t}\hat{u}_\theta(x,t) + \mathcal{F}_x\hat{u}_\theta(x,t) - r_i\|^2 + \frac{1}{N_d}\sum_{i=1}^{N_d} \|\hat{u}_\theta(x_i,t_i) - u_i^*\|^2.$$

The gradients in $\mathcal{F}$ are derived using automated differentiation. The resulting predictions are thus driven to inherit any physical attributes imposed by the PDE constraint [191]. The physics constraints are included in the loss function to enforce model training, which can accurately reflect latent system nonlinearity even when training data points are scarce [197].

Observations About the Loss

The loss $\mathcal{L}_{\mathcal{F}}(\theta)$ is calculated by utilizing automated differentiation (AD) to compute the derivatives of $\hat{u}_\theta(z)$ [60]. Most ML libraries, including TensorFlow and Pytorch, provide AD, which is mostly used to compute derivatives with respect to DNN weights (i.e. θ). AD permits the PINN approach to implement any PDE and boundary condition requirements without numerically discretizing and solving the PDE [60].

Additionally, by applying PDE constraints via the penalty term $\mathcal{L}_{\mathcal{F}}(\theta)$, it is possible to use the related weight $\omega_{\mathcal{F}}$ to account for the PDE model's fidelity. To a low-fidelity PDE model, for example, can be given a lower weight. In general, the number of unknown parameters in θ is substantially greater than the number of measurements, therefore regularization is required for DNN training [60].

By removing loss for equations from the optimization process (i.e., setting $\omega_{\mathcal{F}} = 0$), neural networks could be trained without any knowledge of the underlying governing equations. Alternatively, supplying initial and boundary conditions for all dynamical variables would correspond to solving the equations directly with neural networks on a regular basis [106].

While it is preferable to enforce the physics model across the entire domain, the computational cost of estimating and reducing the loss function (6), while training, grows with the number of residual points [60]. Apart the number of residual points, also the position (distribution) of residual points are crucial parameters in PINNs because they can change the design of the loss function [103].

A deep neural network can reduce approximation error by increasing network expressivity, but it can also produce a large generalization error. Other hyperparameters, such as learning rate, number of iterations, and so on, can be adjusted to further control and improve this issue.

The addition of extra parameters layer by layer in a NN modifies the slope of the activation function in each hidden-layer, improving the training speed. Through the slope recovery term, these activation slopes can also contribute to the loss function [71].

而对于数据点，

$$\mathcal{L}_{data}(\theta) = MSE_{data} = \frac{1}{N_d}\sum_{i=1}^{N_d} \|\hat{u}_\theta(z_i) - u_i^*\|^2.$$

计算近似值 $u(t,\quad x)$ 在已知数据点处的误差。在正向问题中，数据损失还可能表示边界和初始条件，而在逆问题中，它指的是域 [82] 内不同位置处的解。

在Raissi和Karniadakis [140], Raissi等人 [146], 原始方法中，总损失（6）被定义为

$$\mathcal{L}(\theta) = \frac{1}{N_c}\sum_{i=1}^{N_c} \|\frac{\partial}{\partial t}\hat{u}_\theta(x,t) + \mathcal{F}_x\hat{u}_\theta(x,t) - r_i\|^2 + \frac{1}{N_d}\sum_{i=1}^{N_d} \|\hat{u}_\theta(x_i,t_i) - u_i^*\|^2.$$

在 $\mathcal{F}$ 中使用自动微分推导梯度。因此，生成的预测结果会继承由PDE约束 [191] 施加的任何物理属性。物理约束被包含在损失函数中，以强制模型训练，这可以在训练数据点稀缺 [197] 的情况下准确反映潜在系统非线性。

关于损失的观察

损失 $\mathcal{L}_{\mathcal{F}}(\theta)$ 是通过利用自动微分 (AD) 来计算 $\hat{}u_\theta(z)$ [60] 的导数来计算的。大多数机器学习库，包括 TensorFlow 和 PyTorch，都提供 AD，它主要用于计算相对于 DNN 权重（即 θ）的导数。AD 允许 PINN 方法在不数值离散化和求解 PDE [60] 的情况下实现任何 PDE 和边界条件要求。

此外，通过通过惩罚项 $\mathcal{L}_{\mathcal{F}}(\theta)$ 应用 PDE 约束，可以使用相关的权重 $\omega_{\mathcal{F}}$ 来考虑 PDE 模型的保真度。例如，对于低保真 PDE 模型，可以给予较低的权重。通常，θ 中未知参数的数量远大于测量值数量，因此需要对 DNN 训练 [60] 进行正则化。

通过从优化过程中移除方程的损失（即设置 $\omega_{\mathcal{F}} = 0$），神经网络可以在没有任何底层控制方程知识的情况下进行训练。或者，为所有动力学变量提供初始和边界条件将对应于定期使用神经网络直接求解方程 [106]。

虽然在整个域中强制执行物理模型是首选的，但在训练过程中估计和减少损失函数（6）的计算成本随着残差点数量的增加而增长 [60]。除了残差点数量，残差点的位置（分布）也是 PINNs 中的关键参数，因为它们可以改变损失函数的设计 [103]。

深度神经网络可以通过提高网络表达能力来减少近似误差，但它也可能产生较大的泛化误差。其他超参数，如学习率、迭代次数等，可以调整以进一步控制和改进这个问题。

在神经网络中逐层添加额外参数会改变每个隐藏层的激活函数的斜率，从而提高训练速度。通过斜率恢复项，这些激活斜率也可以对损失函数 [71] 贡献。

Soft and Hard Constraint

BC constraints can be regarded as penalty terms (soft BC enforcement) [200], or they can be encoded into the network design (hard BC enforcement) [168]. Many existing PINN frameworks use a *soft* approach to constrain the BCs by creating extra loss components defined on the collocation points of borders. The disadvantages of this technique are twofold:

1. satisfying the BCs accurately is not guaranteed;
2. the assigned weight of BC loss might effect learning efficiency, and no theory exists to guide determining the weights at this time.

Zhu et al [199] address the Dirichlet BC in a *hard* approach by employing a specific component of the neural network to purely meet the specified Dirichlet BC. Therefore, the initial boundary conditions are regarded as part of the labeled data constraint.

When compared to the residual-based loss functions typically found in the literature, the variational energy-based loss function is simpler to minimize and so performs better [52]. Loss function can be constructed using collocation points, weighted residuals derived by the Galerkin–Method [76], or energy based. Alternative loss functions approaches are compared in Li et al [94], by using either only data-driven (with no physics model), a PDE-based loss, and an energy-based loss. They observe that there are advantages and disadvantages for both PDE-based and energy-based approaches. PDE-based loss function has more hyperparameters than the energy-based loss function. The energy-based strategy is more sensitive to the size and resolution of the training samples than the PDE-based strategy, but it is more computationally efficient.

Optimization Methods

The minimization process of the loss function is called *training*; in most of the PINN literature, loss functions are optimized using minibatch sampling using Adam and the limited-memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS) algorithm, a quasi-Newton optimization algorithm. When monitoring noisy data, Mathews et al [106] found that increasing the sample size and training only with L-BFGS achieved the optimum for learning.

For a moderately sized NN, such as one with four hidden layers (depth of the NN is 5) and twenty neurons in each layer (width of the NN is 20), we have over 1000 parameters to optimize. There are several local minima for the loss function, and the gradient-based optimizer will almost certainly become caught in one of them; finding global minima is an NP-hard problem [128].

The Adam approach, which combines adaptive learning rate and momentum methods, is employed in Zhu et al [199] to increase convergence speed, because stochastic gradient descent (SGD) hardly manages random collocation points, especially in 3D setup.

Yang et al [189] use Wasserstein GANs with gradient penalty (WGAN–GP) and prove that they are more stable than vanilla GANs, in particular for approximating stochastic processes with deterministic boundary conditions.

cPINN [71] allow to flexibly select network hyper-parameters such as optimization technique, activation function, network depth, or network width based on intuitive knowledge of solution regularity in each sub-domain. E.g. for smooth zones, a shallow network may be used, and a deep neural network can be used in an area where a complex nature is assumed.

He et al [60] propose a two-step training approach in which the loss function is minimized first by the Adam algorithm with a predefined stop condition, then by the L-BFGS-B optimizer. According to the aforementioned paper, for cases with a little amount of training

软硬约束

边界条件约束可以被看作是惩罚项（软边界条件执行） [200], ，或者可以被编码到网络设计中（硬边界条件执行） [168]。许多现有的物理信息神经网络框架使用 软 方法通过在边界配置点上定义额外的损失组件来约束边界条件。这种技术的缺点有两个：

1. 无法保证精确满足边界条件；2. 边界条件损失的分配权重可能影响学习效率，目前还没有理论来指导确定权重。

朱等人 [199] 通过采用神经网络的一个特定组件来以 硬 方法处理狄利克雷边界条件，从而纯粹满足指定的狄利克雷边界条件。因此，初始边界条件被视为标记数据约束的一部分。

与文献中常见的基于残差的损失函数相比，变分能量损失函数更易于最小化，因此表现更好 [52]。损失函数可以使用配置点、伽辽金方法 [76]，导出的加权残差或基于能量的方式构建。Liet al [94]，通过使用仅数据驱动（无物理模型）、基于偏微分方程的损失和基于能量的损失等方法比较了替代损失函数方法。他们观察到基于偏微分方程和基于能量的方法都有其优势和劣势。基于偏微分方程的损失函数比基于能量的损失函数具有更多的超参数。基于能量的策略比基于偏微分方程的策略对训练样本的大小和分辨率更敏感，但它更具有计算效率。

优化方法

损失函数的最小化过程称为训练；在大多数物理信息神经网络文献中，损失函数使用Adam 和限制内存 Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) 算法（一种拟牛顿优化算法）进行优化。当监控噪声数据时，Mathews 等人 [106] 发现增加样本大小并仅使用 L-BFGS 可以达到学习的最优。

对于一个中等规模的神经网络，例如具有四个隐藏层（网络深度为5）和每层二十个神经元（神经网络的宽度为20）的网络，我们有超过1000个需要优化的参数。损失函数存在多个局部最小值，基于梯度的优化器几乎肯定会陷入其中一个；寻找全局最小值是一个NP难问题 [128]。

Adam方法结合了自适应学习率和动量方法，在朱等人 [199] 中被采用以提高收敛速度，因为随机梯度下降（SGD）几乎无法管理随机配置点，尤其是在3D设置中。

杨等人 [189] 使用Wasserstein GANs with gradient penalty (WGAN–GP) 并证明它们比原味GANs更稳定，特别是在逼近具有确定性边界条件的随机过程时。

cPINN [71] 允许根据每个子域中解的规律性的直观知识灵活选择网络超参数，例如优化技术、激活函数、网络深度或网络宽度。例如，对于平滑区域，可以使用浅层网络，而在假设具有复杂性质的区域可以使用深度神经网络。

何等等人 [60] 提出了一种两步训练方法，首先通过Adam算法以预定义停止条件最小化损失函数，然后通过L-BFGS-B优化器进行优化。根据上述论文，对于训练数据量较少的情况

data and/or residual points, L-BFGS-B, performs better with a faster rate of convergence and reduced computing cost.

Finally, let's look at a practical examples for optimizing the training process; dimensionless and normalized data are used in DeepONet training to improve stability [96]. Moreover the governing equations in dimensionless form, including the Stokes equations, electric potential, and ion transport equations, are presented in DeepM&Mnets [21].

In terms of training procedure initialization, slow and fast convergence behaviors are produced by bad and good initialization, respectively, but Pang et al [128] reports a technique for selecting the most suitable one. By using a limited number of iterations, one can first solve the inverse problem. This preliminary solution has low accuracy due to discretization, sampling, and optimization errors. The optimized parameters from the low-fidelity problem can then be used as a suitable initialization.

2.4 Learning Theory of PINN

This final subsection provides most recent theoretical studies on PINN to better understand how they work and their potential limits. These investigations are still in their early stages, and much work remains to be done.

Let us start by looking at how PINN can approximate the true solution of a differential equation, similar to how error analysis is done a computational framework. In traditional numerical analysis, we approximate the true solution $u(z)$ of a problem with an approximation scheme that computes $\hat{u}_\theta(z)$. The main theoretical issue is to estimate the global error

$$\mathcal{E} = \hat{u}_\theta(z) - u(z).$$

Ideally, we want to find a set of parameters, θ, such that $\mathcal{E} = 0$.

When solving differential equations using a numerical discretization technique, we are interested in the numerical method's stability, consistency, and convergence [8, 150, 171]. In such setting, discretization's error can be bound in terms of consistency and stability, a basic result in numerical analysis. The Lax–Richtmyer Equivalence Theorem is often referred to as a fundamental result of numerical analysis where roughly the convergence is ensured when there is consistency and stability.

When studying PINN to mimic this paradigm, the convergence and stability are related to how well the NN learns from physical laws and data. In this conceptual framework, we use a NN, which is a parameterized approximation of problem solutions modeled by physical laws. In this context, we will (i) introduce the concept of convergence for PINNs, (ii) revisit the main error analysis definitions in a statistical learning framework, and (iii) finally report results for the generalization error.

2.4.1 Convergence Aspects

The goal of a mathematical foundation for the PINN theory is to investigate the convergence of the computed $\hat{u}_\theta(z)$ to the solution of problem (1), $u(z)$.

Consider a NN configuration with coefficients compounded in the vector θ and a cardinality equal to the number of coefficients of the NN, $\#\theta$. In such setting, we can consider the hypothesis class

$$\mathcal{H}_n = \{u_\theta : \#\theta = n\}$$

数据以及/或残差点，L-BFGS-B，收敛速度更快且计算成本更低。

最后，让我们看一下优化训练过程的实际示例；DeepONet训练中使用无量纲和归一化数据以提高稳定性 [96]。此外，无量纲形式下的控制方程，包括斯托克斯方程、电势和离子传输方程，在DeepM&Mnets [21]中给出。

在训练程序初始化方面，不良和良好初始化分别产生慢收敛和快收敛行为，但Pang等人 [128] 报告了一种选择最合适的方法的技术。通过使用有限的迭代次数，可以首先求解逆问题。由于离散化、采样和优化误差，这个初步解的精度较低。然后，可以从低保真问题中获得的优化参数用作合适的初始化。

2.4 PINN的学习理论

本节最后的小节提供了关于物理信息神经网络（PINN）的最新理论研究，以更好地理解它们的工作原理及其潜在限制。这些研究仍处于早期阶段，还有大量工作需要完成。

让我们从研究PINN如何逼近微分方程的真实解开始，类似于计算框架中的误差分析。在传统的数值分析中，我们使用一种近似方案来计算问题中的真实解$u(z)$，该方案计算 $\hat{}u_\theta(z)$。主要的理论问题是估计全局误差

$$\mathcal{E} = \hat{u}_\theta(z) - u(z).$$

理想情况下，我们希望找到一组参数， θ，使得$\mathcal{E} = 0$。

当使用数值离散化技术求解微分方程时，我们关注数值方法的稳定性、一致性和收敛性 [8, 150, 171]。在这种情况下，离散化的误差可以用一致性和稳定性来界定，这是数值分析中的一个基本结果。Lax-Richtmyer等价定理通常被称为数值分析的一个基本结果，其中大致上，当存在一致性和稳定性时，收敛性得到了保证。

当研究物理信息神经网络（PINN）以模拟这一范式时，收敛性和稳定性与神经网络从物理定律和数据中学习的效果有关。在这个概念框架中，我们使用一个神经网络，它是物理定律建模的问题解的参数化近似。在这种情况下，我们将（i）介绍物理信息神经网络的收敛性概念，（ii）在统计学习框架中重新审视主要的误差分析定义，以及（iii）最终报告泛化误差的结果。

2.4.1 收敛性方面

PINN理论数学基础的目标是研究计算结果 $\hat{}u_\theta(z)$ 收敛到问题（1）的解$u(z)$。

考虑一个神经网络配置，其系数在向量 θ 中组合，并且基数等于神经网络系数的数量，$\#\theta$。在这种情况下，我们可以考虑假设类

$$\mathcal{H}_n = \{u_\theta : \#\theta = n\}$$

composed of all the predictors representing a NN whose number of coefficients of the architecture is n. The capacity of a PINN to be able to learn, is related to how big is n, i.e. the expressivity of $\mathcal{H}_n$.

In such setting, a theoretical issue, is to investigate, how dose the sequence of compute predictors, $\hat{u}_\theta$, converges to the solution of the physical problem (1)

$$\hat{u}_{\theta(n)} \to u, \qquad n \to \infty.$$

A recent result in this direction was obtained by De Ryck et al [37] in which they proved that the difference $\hat{u}_{\theta(n)} - u$ converges to zero as the width of a predefined NN, with activation function tanh, goes to infinity.

Practically, the PINN requires choosing a network class $\mathcal{H}_n$ and a loss function given a collection of N-training data [157]. Since the quantity and quality of training data affect $\mathcal{H}_n$, the goal is to minimize the loss, by finding a $u_{\theta^*} \in \mathcal{H}_n$, by training the N using an optimization process. Even if $\mathcal{H}_n$ includes the exact solution u to PDEs and a global minimizer is established, there is no guarantee that the minimizer and the solution u will coincide. A first work related on PINN [157], the authors show that the sequence of minimizers $\hat{u}_{\theta^*}$ strongly converges to the solution of a linear second-order elliptic and parabolic type PDE.

2.4.2 Statistical Learning Error Analysis

The entire learning process of a PINN can be considered as a statistical learning problem, and it involves mathematical foundations aspects for the error analysis [88]. For a mathematical treatment of errors in PINN, it is important to take into account: optimization, generalization errors, and approximation error. It is worth noting that the last one is dependent on the architectural design.

Let be N collocation points on $\bar{\Omega} = \Omega \cup \partial\Omega$, a NN approximation realized with θ, denoted by $\hat{u}_\theta$, evaluated at points z_i, whose exact value is h_i. Following the notation of Kutyniok [88], the *empirical risk* is defined as

$$\widehat{\mathcal{R}}[u_\theta] := \frac{1}{N}\sum_{i=1}^{N} \|\hat{u}_\theta(z_i) - h_i,\|^2 \tag{7}$$

and represents how well the NN is able to predict the exact value of the problem. The empirical risk actually corresponds to the loss defined in 2.3, where $\hat{u}_\theta = \mathcal{F}(\hat{u}_\theta(z_i))$ and $h_i = f(z_i)$ and similarly for the boundaries.

A continuum perspective is the *risk* of using an approximator $\hat{u}_\theta$, calculated as follows:

$$\mathcal{R}[\hat{u}_\theta] := \int_{\bar{\Omega}} (\hat{u}_\theta(z) - u(z))^2 \, dz, \tag{8}$$

where the distance between the approximation $\hat{u}_\theta$ and the solution u is obtained with the L^2-norm. The final approximation computed by the PINN, after a training process of DNN, is $\hat{u}^*_\theta$. The main aim in error analysis, is to find suitable estimate for the risk of predicting u i.e. $\mathcal{R}[\hat{u}^*_\theta]$.

The training process, uses gradient-based optimization techniques to minimize a generally non convex cost function. In practice, the algorithmic optimization scheme will not always find a global minimum. So the error analysis takes into account the *optimization error* defined as follows:

$$\mathcal{E}_O := \widehat{\mathcal{R}}[\hat{u}^*_\theta] - \inf_{\theta\in\Theta} \widehat{\mathcal{R}}[u_\theta]$$

由所有代表一个神经网络的所有预测变量组成，该神经网络的架构系数数量是 n。PINN能够学习的能力与其相关，即 $\mathcal{H}_n$的表达能力如何大。

在这种情况下，一个理论问题，是研究，计算预测变量的序列 $\hat{u}_\theta$，收敛到物理问题的解 (1)

$$\hat{u}_{\theta(n)} \to u, \qquad n \to \infty.$$

在这个方向上最近的结果是由De Ryck等人 [37] 获得的，他们证明了当具有tanh激活函数的预定义神经网络的宽度趋于无穷大时，差异 $\hat{u}_{\theta(n)} - u$ 收敛到零。

实际上，PINN需要根据一组 N -训练数据 [157]选择网络类别 $\mathcal{H}_n$ 和损失函数。由于训练数据的数量和质量影响$\mathcal{H}_n$，目标是通过对 N 进行优化过程训练，找到$u_{\theta^*} \in \mathcal{H}_n$，以最小化损失。即使$\mathcal{H}_n$包括PDEs的精确解 u 并建立了全局最小值，也没有保证最小值和解决方案u 会一致。一项与PINN相关的工作 [157], ，作者表明，最小值序列 $\hat{u}_{\theta^*}$ 强收敛到线性二阶椭圆和抛物型PDE的解。

2.4.2 Statistical Learning Error Analysis

一个PINN的整个学习过程可以被视为一个统计学习问题，并且它涉及误差分析的数学基础方面 [88]。对于PINN中误差的数学处理，重要的是要考虑：优化、泛化误差和近似误差。值得注意的是，最后一个是依赖于架构设计的。

设 $\bar{\Omega} = \Omega \cup \partial\Omega$上有$N$ 个配置点，一个用 θ实现的神经网络近似，记为 $\hat{u}_\theta$，在点z_i处被评估，其精确值为h_i。根据库蒂诺克[88], 的符号，经验风险定义为

$$\widehat{\mathcal{R}}[u_\theta] := \frac{1}{N}\sum_{i=1}^{N} \|\hat{u}_\theta(z_i) - h_i,\|^2 \tag{7}$$

和表示神经网络预测问题精确值的能力如何。实际上，经验风险对应于在2.3中定义的损失，其中 $\hat{u}_\theta = \mathcal{F}(\hat{u}_\theta(z_i))$ 和$h_i = f(z_i)$，边界情况类似。

连续体视角是使用近似器 $\hat{u}_\theta$ 的 风险 ，计算方法如下：

$$\mathcal{R}[\hat{u}_\theta] := \int_{\bar{\Omega}} (\hat{u}_\theta(z) - u(z))^2 \, dz, \tag{8}$$

其中，近似 $\hat{u}_\theta$ 与解 u 之间的距离通过 $L2$-范数获得。PINN 在经过 DNN 的训练过程后计算出的最终近似值为 $\hat{u}^*_\theta$。误差分析的主要目标是为预测 u 的风险找到合适的估计值，即 $\mathcal{R}[\hat{u}^*_\theta]$。

训练过程使用基于梯度的优化技术来最小化一个通常是非凸的成本函数。在实践中，算法优化方案并不总是能找到全局最小值。因此，误差分析考虑了如下定义的 优化误差 ：

$$\mathcal{E}_O := \widehat{\mathcal{R}}[\hat{u}^*_\theta] - \inf_{\theta\in\Theta} \widehat{\mathcal{R}}[u_\theta]$$

Because the objective function is nonconvex, the optimization error is unknown. Optimization frequently involves a variety of engineering methods and time-consuming fine-tuning, using, gradient-based optimization methods. Several stochastic gradient descent methods have been proposed,and many PINN use Adam and L-BFGS. Empirical evidence suggests that gradient-based optimization techniques perform well in different challenging tasks; however, gradient-based optimization might not find a global minimum for many ML tasks, such as for PINN, and this is still an open problem [157].

Moreover a measure of the prediction accuracy on unseen data in machine learning is the *generalization error*:

$$\mathcal{E}_G := \sup_{\theta\in\Theta} |\mathcal{R}[u_\theta] - \widehat{\mathcal{R}}[u_\theta]|$$

The generalization error measures how well the loss integral is approximated in relation to a specific trained neural network. One of the first paper focused with convergence of generalization error is Shin et al [157].

About the ability of the NN to approximate the exact solution, the *approximation error* is defined as

$$\mathcal{E}_A := \inf_{\theta\in\Theta} \mathcal{R}[u_\theta]$$

The approximation error is well studied in general, in fact we know that one layer neural network with a high width can evenly estimate a function and its partial derivative as shown by Pinkus [133].

Finally, as stated in Kutyniok [88], the global error between the trained deep neural network $\hat{u}^*_\theta$ and the correct solution function u of problem (1), can so be bounded by the previously defined error in the following way

$$\mathcal{R}[\hat{u}^*_\theta] \leq \mathcal{E}_O + 2\mathcal{E}_G + \mathcal{E}_A \tag{9}$$

These considerations lead to the major research threads addressed in recent studies, which are currently being investigated for PINN and DNNs in general Kutyniok [88].

2.4.3 Error Analysis Results for PINN

About the approximating error, since it depends on the NN architecture, mathematical foundations results are generally discussed in papers deeply focused on this topic Calin [24], Elbrächter et al [43].

However, a first argumentation strictly related to PINN is reported in Shin et al [158]. One of the main theoretical results on $\mathcal{E}_A$, can be found in De Ryck et al [37]. They demonstrate that for a neural network with a tanh activation function and only two hidden layers, $\hat{u}_\theta$, may approximate a function u with a bound in a Sobolev space:

$$\|\hat{u}_{\theta_N} - u\|_{W^{k,\infty}} \leq C\frac{\ln(cN)^k}{N^{s-k}}$$

where N is the number of training points, $c, C > 0$ are constants independent of N and explicitly known, $u \in W^{s,\infty}([0,1]^d)$. We remark that the NN has width N^d, and $\#\theta$ depends on both the number of training points N and the dimension of the problem d.

Formal findings for generalization errors in PINN are provided specifically for a certain class of PDE. In Shin et al [157] they provide convergence estimate for linear second-order elliptic and parabolic type PDEs, while in Shin et al [158] they extend the results to

由于目标函数是非凸的，优化误差是未知的。优化通常涉及各种工程方法以及耗时较长的微调，使用基于梯度的优化方法。已经提出了几种随机梯度下降方法，许多物理信息神经网络使用Adam和L-BFGS。经验证据表明，基于梯度的优化技术在不同的挑战性任务中表现良好；然而，基于梯度的优化可能不会为许多机器学习任务找到一个全局最小值，例如对于物理信息神经网络，而这仍然是一个开放的问题 [157]。

此外，机器学习中未见数据上的预测精度的一种度量是泛化误差：

$$\mathcal{E}_G := \sup_{\theta\in\Theta} |\mathcal{R}[u_\theta] - \widehat{\mathcal{R}}[u_\theta]|$$

泛化误差衡量损失积分相对于特定训练的神经网络近似得有多好。最早关注泛化误差收敛性的一篇论文是Shin等人 [157]。

关于神经网络近似精确解的能力，近似误差被定义为

$$\mathcal{E}_A := \inf_{\theta\in\Theta} \mathcal{R}[u_\theta]$$

逼近误差在一般情况下得到了深入研究，事实上，我们知道一个具有高宽度的单层神经网络可以均匀地估计一个函数及其偏导数，如Pinkus [133]所示。

最后，如Kutyniok [88],训练的深度神经网络$\hat{u}^*_\theta$与问题（1）的正确解函数 u 之间的全局误差可以如此被先前定义的误差以以下方式所界定

$$\mathcal{R}[\hat{u}^*_\theta] \leq \mathcal{E}_O + 2\mathcal{E}_G + \mathcal{E}_A \tag{9}$$

这些考虑导致了最近研究中解决的主要研究线索，这些线索目前正针对PINN和深度神经网络（DNNs）进行深入研究 Kutyniok [88]。

2.4.3 PINN的误差分析结果

关于逼近误差，由于它取决于神经网络（NN）架构，数学基础结果通常在深入关注该主题的论文中讨论 Calin [24],Elbrächter等人 [43]。

然而，Shin等人 [158]报告了一个与PINN严格相关的主要理论结果。$\mathcal{E}_A$的主要理论结果之一可以在De Ryck等人 [37]中找到。他们证明，对于一个具有tanh激活函数且只有两层隐藏层的神经网络，$\hat{u}_\theta$可以在Sobolev空间中对函数 u 进行逼近，并具有界限：

$$\|\hat{u}_{\theta_N} - u\|_{W^{k,\infty}} \leq C\frac{\ln(cN)^k}{N^{s-k}}$$

其中 N 表示训练点的数量，$c,C > 0$ 是独立于 N 且明确已知的常数，$u \in W_{s,\infty}([0,1]^d)$。我们指出，该神经网络（NN）的宽度为 N^d，而 $\#\theta$则取决于训练点的数量 N 以及问题的维度d。

关于PINN泛化误差的正式结论特指某一类偏微分方程（PDE）。在Shin等人 [157] 的研究中，他们为线性二阶椭圆型和抛物型偏微分方程提供了收敛估计，而在Shin等人 [158] 的研究中，他们进一步将结果扩展到

all linear problems, including hyperbolic equations. Mishra and Molinaro [113] gives an abstract framework for PINN on forward problem for PDEs, they estimate the generalization error by means of training error (empirical risk), and number of training points, such abstract framework is also addressed for inverse problems [111]. In De Ryck et al [38] the authors specifically address Navier–Stokes equations and show that small training error imply a small generalization error, by proving that

$$\mathcal{R}[\hat{u}_\theta] = \|u - \hat{u}_\theta\|_{L^2} \le \left(C\widehat{\mathcal{R}}[u_\theta] + \mathcal{O}\left(N^{-\frac{1}{d}}\right)\right)^{\frac{1}{2}}.$$

This estimate suffer from the curse of dimensionality (CoD), that is to say, in order to reduce the error by a certain factor, the number of training points needed and the size of the neural network, scales up exponentially.

Finally, a recent work [18] has proposed explicit error estimates and stability analyses for incompressible Navier–Stokes equations.

De Ryck and Mishra [36] prove that for a Kolmogorov type PDE (i.e. heat equation or Black–Scholes equation), the following inequality holds, almost always,

$$\mathcal{R}[\hat{u}_\theta] \le \left(C\widehat{\mathcal{R}}[u_\theta] + \mathcal{O}\left(N^{-\frac{1}{2}}\right)\right)^{\frac{1}{2}},$$

and is not dependant on the dimension of the problem d.

Finally, Mishra and Molinaro [112] investigates the radiative transfer equation, which is noteworthy for its high-dimensionality, with the radiative intensity being a function of 7 variables (instead of 3, as common in many physical problems). The authors prove also here that the generalization error is bounded by the training error and the number of training points, and the dimensional dependence is on a logarithmic factor:

$$\mathcal{R}[\hat{u}_\theta] \le \left(C\widehat{\mathcal{R}}[u_\theta]^2 + c\left(\frac{(\ln N)^{2d}}{N}\right)\right)^{\frac{1}{2}}.$$

The authors are able to show that PINN does not suffer from the dimensionality curse for this problem, observing that the training error does not depend on the dimension but only on the number of training points.

3 Differential Problems Dealt with PINNs

The first vanilla PINN [146] was built to solve complex nonlinear PDE equations of the form $u_t + \mathcal{F}_x u = 0$, where x is a vector of space coordinates, t is a vector time coordinate, and $\mathcal{F}_x$ is a nonlinear differential operator with respect to spatial coordinates. First and mainly, the PINN architecture was shown to be capable of handling both forward and inverse problems. Eventually, in the years ahead, PINNs have been employed to solve a wide variety of ordinary differential equations (ODEs), partial differential equations (PDEs), Fractional PDEs, and integro-differential equations (IDEs), as well as stochastic differential equations (SDEs). This section is dedicated to illustrate where research has progressed in addressing various sorts of equations, by grouping equations according to their form and addressing the primary work in literature that employed PINN to solve such equation. All PINN papers dealing with ODEs will be presented first. Then, works on steady-state PDEs such as Elliptic type equations, steady-state diffusion, and the Eikonal equation are reported. The Navier–Stokes equations are followed by more dynamical problems such as heat transport, advection-diffusion-reaction system, hyperbolic equations, and Euler equations or quantum harmonic

所有线性问题，包括双曲型方程。Mishra和Molinaro [113]gives an abstract framework for PINN on forward problem for PDEs, they estimate the generalization error by means of training error (经验风险), and number of training points, such abstract framework is also addressed for inverse problems [111]. In De Ryck et al [38] the authors specifically address Navier–Stokes equations and showthat small training error imply a small generalization error, by proving that

$$\mathcal{R}[\hat{u}_\theta] = \|u - \hat{u}_\theta\|_{L^2} \le \left(C\widehat{\mathcal{R}}[u_\theta] + \mathcal{O}\left(N^{-\frac{1}{d}}\right)\right)^{\frac{1}{2}}.$$

这种估计受到维数灾难（CoD）的影响，也就是说，为了将误差降低一定比例，所需的训练点数量和神经网络的规模会呈指数级增长。

最后，一项近期工作 [18] 提出了不可压缩纳维-斯托克斯方程的显式误差估计和稳定性分析。

De Ryck and Mishra [36] prove that for a Kolmogorov type PDE (i.e. 热方程 or 布莱克-斯科尔斯方程), thefollowing inequality holds, almost always,

$$\mathcal{R}[\hat{u}_\theta] \le \left(C\widehat{\mathcal{R}}[u_\theta] + \mathcal{O}\left(N^{-\frac{1}{2}}\right)\right)^{\frac{1}{2}},$$

并且不依赖于问题的维度d.

最后，Mishra和Molinaro [112] 研究了辐射传输方程，该方程因其高维性而值得注意，辐射强度是7个变量的函数（而不是像许多物理问题中常见的3个变量）。作者们在这里也证明了泛化误差被训练误差和训练点数所限制，维度依赖性是一个对数因子：

$$\mathcal{R}[\hat{u}_\theta] \le \left(C\widehat{\mathcal{R}}[u_\theta]^2 + c\left(\frac{(\ln N)^{2d}}{N}\right)\right)^{\frac{1}{2}}.$$

作者们能够证明，对于这个问题，PINN不会受到维度诅咒的影响，观察到训练误差不依赖于维度，而只依赖于训练点数。

3 Differential Problems Dealt with PINNs

第一个基础物理信息神经网络 [146] 被构建用来求解形式为 $u_t + \mathcal{F}_x u = 0$ 的复杂非线性偏微分方程，其中 x 是空间坐标的向量，t 是时间坐标的向量，而 $\mathcal{F}_x$ 是关于空间坐标的非线性微分算子。首先和主要地，PINN架构被证明能够处理正向和逆向问题。最终，在未来几年中，PINNs已被用于求解各种常微分方程（ODEs）、偏微分方程（PDEs）、分数阶偏微分方程以及积分微分方程（IDEs），以及随机微分方程（SDEs）。本节致力于说明在处理各种方程方面研究进展，通过按方程形式分组并介绍主要使用PINN求解此类方程的文献中的工作。所有处理ODEs的PINN论文将首先介绍。然后，将报告关于稳态PDEs的工作，如椭圆型方程、稳态扩散和艾克顿方程。Navier–Stokes方程随后是更动态的问题，如热传输、对流-扩散-反应系统、双曲型方程以及欧拉方程或量子谐波

oscillator. Finally, while all of the previous PDEs can be addressed in their respective Bayesian problems, the final section provides insight into how uncertainly is addressed, as in stochastic equations.

3.1 Ordinary Differential Equations

ODEs can be used to simulate complex nonlinear systems which are difficult to model using simply physics-based models [91]. A typical ODE system is written as

$$\frac{du(x,t)}{dt} = f(u(x,t),t)$$

where initial conditions can be specified as $\mathcal{B}(u(t)) = g(t)$, resulting in an initial value problem or boundary value problem with $\mathcal{B}(u(x)) = g(x)$. A PINN approach is used by Lai et al [91] for structural identification, using Neural Ordinary Differential Equations (Neural ODEs). Neural ODEs can be considered as a continuous representation of ResNets (Residual Networks), by using a neural network to parameterize a dynamical system in the form of ODE for an initial value problem (IVP):

$$\frac{du(t)}{dt} = f_\theta(u(t),t);\, u(t_0) = u_0$$

where f is the neural network parameterized by the vector θ.

The idea is to use Neural ODEs as learners of the governing dynamics of the systems, and so to structure of Neural ODEs into two parts: a physics-informed term and an unknown discrepancy term. The framework is tested using a spring-mass model as a 4-degree-of-freedom dynamical system with cubic nonlinearity, with also noisy measured data. Furthermore, they use experimental data to learn the governing dynamics of a structure equipped with a negative stiffness device [91].

Zhang et al [197] employ deep long short-term memory (LSTM) networks in the PINN approach to solve nonlinear structural system subjected to seismic excitation, like steel moment resistant frame and the single degree-of-freedom Bouc–Wen model, a nonlinear system with rate-dependent hysteresis [197]. In general they tried to address the problems of nonlinear equation of motion :

$$\ddot{\mathbf{u}} + \mathbf{g} = -\boldsymbol{\Gamma} a_g$$

where $\mathbf{g}(t) = \mathbf{M}^{-1}\mathbf{h}(t)$ denotes the mass-normalized restoring force, being $\mathbf{M}$ the mass matrices; $\mathbf{h}$ the total nonlinear restoring force, and $\boldsymbol{\Gamma}$ force distribution vector.

Directed graph models can be used to directly implement ODE as deep neural networks [174], while using an Euler RNN for numerical integration.

In Nascimento et al [121] is presented a tutorial on how to use Python to implement the integration of ODEs using recurrent neural networks.

ODE-net idea is used in Tong et al [172] for creating Symplectic Taylor neural networks. These NNs consist of two sub-networks, that use symplectic integrators instead of Runge-Kutta, as done originally in ODE-net, which are based on residual blocks calculated with the Euler method. Hamiltonian systems as Lotka–Volterra, Kepler, and Hénon–Heiles systems are also tested in the aforementioned paper.

振荡器。最后，虽然所有之前的偏微分方程 (PDEs) 都可以在各自的贝叶斯问题 (Bayesian problems) 中解决，但最后一节提供了关于如何处理不确定性的见解，如在随机方程 (stochastic equations) 中。

3.1 Ordinary Differential Equations

常微分方程 (ODEs) 可用于模拟复杂非线性系统，这些系统难以仅使用基于物理的模型 (physics-based models) [91] 来建模。一个典型的 ODE 系统可以写成

$$\frac{du(x,t)}{dt} = f(u(x,t),t)$$

其中初始条件 (initial conditions) 可以指定为$\mathcal{B}(u(t)) = g(t)$，从而得到一个初值问题 (initial value problem) 或边值问题 (boundary value problem)，具有 $\mathcal{B}(u(x)) = g(x)$。李等人 (Lai et al [91]) 使用物理信息神经网络 (PINN) 方法 (PINN approach) 进行结构识别 (structural identification)，使用神经常微分方程 (Neural Ordinary Differential Equations) (Neural ODEs)。神经常微分方程可以被视为残差网络 (ResNets (Residual Networks)) 的连续表示，通过使用神经网络来参数化一个动力系统 (dynamical system)，该动力系统以常微分方程 (ODE) 的形式表示初值问题 (IVP)：

$$\frac{du(t)}{dt} = f_\theta(u(t),t);\, u(t_0) = u_0$$

其中 f 是由向量 θ 参数化的神经网络。

其思想是使用神经常微分方程（Neural ODEs）作为系统主导动力学的学习器，并将神经常微分方程的结构分为两部分：物理信息项和未知差异项。该框架使用弹簧质量模型作为具有三次非线性的四自由度动力系统进行测试，并使用噪声测量数据。此外，他们使用实验数据来学习配备负刚度装置的结构的主导动力学 [91]。

张等人 [197] 在物理信息神经网络（PINN）方法中采用深度长短期记忆（LSTM）网络来求解受地震激励的非线性结构系统，如钢矩抵抗框架和单自由度Bouc–Wen模型，这是一个具有速率相关迟滞的非线性系统 [197]。通常他们试图解决非线性运动方程的问题：

$$\ddot{\mathbf{u}} + \mathbf{g} = -\boldsymbol{\Gamma} a_g$$

其中 $\mathbf{g}(t) = \mathbf{M}^{-1}\mathbf{h}(t)$ 表示质量归一化恢复力，其中 $\mathbf{M}$ 是质量矩阵；$\mathbf{h}$ 是总非线性恢复力，以及 $\boldsymbol{\Gamma}$ 力分布向量。

有向图模型可直接将常微分方程实现为深度神经网络[174], ，同时使用欧拉RNN进行数值积分。

在纳西索等人 [121] 中，介绍了一种使用Python通过循环神经网络实现常微分方程积分的方法。

ODE-net的思想在Tong等人 [172] 中用于创建辛泰勒神经网络。这些神经网络由两个子网络组成，它们使用辛积分器代替龙格-库塔方法，正如ODE-net最初所做的那样，该方法是使用欧拉方法计算的残差块。上述论文还测试了哈密顿系统，如洛特卡-沃尔泰拉、开普勒和赫农-海尔斯系统。

3.2 Partial Differential Equations

Partial Differential Equations are the building bricks of a large part of models that are used to mathematically describe physics phenomenologies. Such models have been deeply investigated and often solved with the help of different numerical strategies. Stability and convergence of these strategies have been deeply investigated in literature, providing a solid theoretical framework to approximately solve differential problems. In this Section, the application of the novel methodology of PINNs on different typologies of Partial Differential models is explored.

3.2.1 Steady State PDEs

In Kharazmi et al [76, 78], a general steady state problem is addressed as:

$$\mathcal{F}_s(u(x); q) = f(x) \quad x \in \Omega,$$
$$\mathcal{B}(u(x)) = 0 \quad x \in \partial\Omega$$

over the domain $\Omega \subset \mathbb{R}^d$ with dimensions d and bounds $\partial\Omega$. $\mathcal{F}_s$ typically contains differential and/or integro-differential operators with parameters q and $f(x)$ indicates some forcing term.

In particular an Elliptic equation can generally be written by setting

$$\mathcal{F}_s(u(x); \sigma, \mu) = -div(\mu \nabla u)) + \sigma u$$

Tartakovsky et al [170] consider a linear

$$\mathcal{F}_s(u(x); \sigma) = \nabla \cdot (K(\mathbf{x}) \nabla u(\mathbf{x})) = 0$$

and non linear

$$\mathcal{F}_s(u(x); \sigma) = \nabla \cdot [K(u) \nabla u(\mathbf{x})] = 0$$

diffusion equation with unknown diffusion coefficient K. The equation essentially describes an unsaturated flow in a homogeneous porous medium, where u is the water pressure and $K(u)$ is the porous medium's conductivity. It is difficult to measure $K(u)$ directly, so Tartakovsky et al [170] assume that only a finite number of measurements of u are available.

Tartakovsky et al [170] demonstrate that the PINN method outperforms the state-of-the-art maximum a posteriori probability method. Moreover, they show that utilizing only capillary pressure data for unsaturated flow, PINNs can estimate the pressure-conductivity for unsaturated flow. One of the first novel approach, PINN based, was the variational physics-informed neural network (VPINN) introduced in Kharazmi et al [76], which has the advantage of decreasing the order of the differential operator through integration-by-parts. The authors tested VPINN with the steady Burgers equation, and on the two dimensional Poisson's equation. VPINN Kharazmi et al [76] is also used to solve Schrödinger Hamiltonians, i.e. an elliptic reaction-diffusion operator [54].

In Haghighat et al [56] a nonlocal approach with the PINN framework is used to solve two-dimensional quasi-static mechanics for linear-elastic and elastoplastic deformation. They define a loss function for elastoplasticity, and the input variables to the feed-forward neural network are the displacements, while the output variables are the components of the strain tensor and stress tensor. The localized deformation and strong gradients in the solution make the boundary value problem difficult solve. The Peridynamic Differential Operator (PDDO) is used in a nonlocal approach with the PINN paradigm in Haghighat et al [56].

3.2 偏微分方程

偏微分方程是用于数学描述物理现象的大型模型的基本构建块。这些模型已被深入研究和使用不同的数值策略求解。文献中已对这些策略的稳定性和收敛性进行了深入研究，为近似求解微分问题提供了坚实的理论基础。在本节中，探索了物理信息神经网络（PINNs）在不同类型的偏微分模型中的应用。

3.2.1 稳态偏微分方程

在哈拉兹米等人 [76, 78]，中，解决了一个一般的稳态问题：

$$\mathcal{F}_s(u(x); q) = f(x) \quad x \in \Omega,$$
$$\mathcal{B}(u(x)) = 0 \quad x \in \partial\Omega$$

在域 $\Omega \subset \mathbb{R}^d$ 上其有维度 d 和边界 $\partial\Omega$。 $\mathcal{F}_s$ 通常包含微分和/或积分微分算子，带有参数 q 和 $f(x)$ 表示某种强制项。

特别是，椭圆方程通常可以通过设置来写出

$$\mathcal{F}_s(u(x); \sigma, \mu) = -div(\mu \nabla u)) + \sigma u$$

Tartakovsky等 [170] 考虑一个线性

$$\mathcal{F}_s(u(x); \sigma) = \nabla \cdot (K(\mathbf{x}) \nabla u(\mathbf{x})) = 0$$

和非线性

$$\mathcal{F}_s(u(x); \sigma) = \nabla \cdot [K(u) \nabla u(\mathbf{x})] = 0$$

扩散方程，其扩散系数未知 K。该方程本质上描述了均匀多孔介质中的非饱和流，其中 u 是水压，$K(u)$ 是多孔介质的导率。直接测量 $K(u)$ 很困难，因此 Tartakovsky等 [170] 假设只有有限数量的 u 测量值可用。

塔塔科夫斯基等 [170]证明了PINN方法优于最先进的最小后验概率方法。此外，他们还表明，仅利用毛细压力数据对非饱和流进行估计，PINNs可以估算非饱和流的压导率。其中一种新颖的方法，基于PINN，是哈拉兹米等人 [76], 引入的变分物理信息神经网络（VPINN），其优点是通过分部积分降低微分算子的阶数。作者们使用VPINN对稳态布格斯方程和二维泊松方程进行了测试。VPINN 哈拉兹米等人 [76] 也被用于求解薛定谔哈密顿量，即椭圆反应扩散算子 [54]。

在哈吉加特等人 [56] 中，使用基于PINN框架的非局部方法求解了线弹性与弹塑性变形的二维准静态力学问题。他们定义了弹塑性损失函数，前馈神经网络的输入变量是位移，而输出变量是应变张量和应力张量的分量。解中的局部变形和强梯度使得边值问题难以求解。哈吉加特等人 [56]在基于PINN范式的非局部方法中使用了周围动力学微分算子（PDDO）。

They demonstrated that the PDDO framework can capture stress and strain concentrations using global functions.

In Dwivedi and Srinivasan [41] the authors address different 1D-2D linear advection and/or diffusion steady-state problems from Berg and Nyström [16], by using their PIELM, a PINN combined with ELM (Extreme Learning Machine). A critical point is that the proposed PIELM only takes into account linear differential operators.

In Ramabathiran and Ramachandran [148] they consider linear elliptic PDEs, such as the solution of the Poisson equation in both regular and irregular domains, by addressing non-smoothness in solutions.

The authors in Ramabathiran and Ramachandran [148] propose a class of partially interpretable sparse neural network architectures (SPINN), and this architecture is achieved by reinterpreting meshless representation of PDE solutions.

Laplace–Beltrami Equation is solved on 3D surfaces, like complex geometries, and high dimensional surfaces, by discussing the relationship between sample size, the structure of the PINN, and accuracy [46].

The PINN paradigm has also been applied to Eikonal equations, i.e. are hyperbolic problems written as

$$\|\nabla u(x)\|^2 = \frac{1}{v^2(x)}, \qquad \forall x \in \Omega,$$

where v is a velocity and u an unknown activation time. The Eikonal equation describes wave propagation, like the travel time of seismic wave [162, 175] or cardiac activation electrical waves [53, 151].

By implementing EikoNet, for solving a 3D Eikonal equation, Smith et al [162] find the travel-time field in heterogeneous 3D structures; however, the proposed PINN model is only valid for a single fixed velocity model, hence changing the velocity, even slightly, requires retraining the neural network. EikoNet essentially predicts the time required to go from a source location to a receiver location, and it has a wide range of applications, like earthquake detection.

PINN is also proved to outperform the first-order fast sweeping solution in accuracy tests [175], especially in the anisotropic model.

Another approach involves synthetic and patient data for learning heart tissue fiber orientations from electroanatomical maps, modeled with anisotropic Eikonal equation [53]. In their implementation the authors add to the loss function also a Total Variation regularization for the conductivity vector.

By neglecting anisotropy, cardiac activation mapping is also addressed by Sahli Costabal et al [151] where PINNs are used with randomized prior functions to quantify data uncertainty and create an adaptive sampling strategy for acquiring and creating activation maps.

Helmholtz equation for weakly inhomogeneous two-dimensional (2D) media under transverse magnetic polarization excitation is addressed in Chen et al [30] as:

$$\nabla^2 E_z(x, y) + \varepsilon_r(x, y)\, k_0^2 E_z = 0,$$

whereas high frequency Helmholtz equation (frequency domain Maxwell's equation) is solved in Fang and Zhan [45].

他们证明了PDDO框架可以使用全局函数捕获应力和应变集中。

在Dwivedi和Srinivasan [41] 中，作者们通过使用他们的PIELM（物理信息神经网络与极限学习机结合）解决了来自Berg和Nyström [16], 的不同1D-2D线性平流和/或扩散稳态问题。一个关键点是，所提出的PIELM只考虑线性微分算子。

在Ramabathiran和Ramachandran [148] 中，他们通过处理解的非光滑性，考虑了线性椭圆偏微分方程，例如泊松方程在规则和不规则区域中的解。

在Ramabathiran和Ramachandran [148] 中的作者们提出了一类部分可解释的稀疏神经网络架构（SPINN），该架构通过重新解释PDE解的无网格表示来实现。

拉普拉斯-贝特拉米方程在三维表面、复杂几何形状和高维表面上进行求解，通过讨论样本大小、PINN的结构和精度之间的关系 [46]。

PINN范式也已被应用于艾克顿方程，即作为双曲问题写成的

$$\|\nabla u(x)\|^2 = \frac{1}{v^2(x)}, \qquad \forall x \in \Omega,$$

其中 v 是速度，u 是一个未知激活时间。艾克顿方程描述了波传播，例如地震波的走时 [162, 175] 或心脏激活电波 [53, 151]。

通过实现EikoNet，用于求解三维艾克顿方程，Smith等人 [162] 在异构三维结构中找到了走时场；然而，所提出的PINN模型仅适用于单一固定速度模型，因此即使稍微改变速度也需要重新训练神经网络。EikoNet本质上预测了从源位置到接收位置所需的时间，并且具有广泛的应用，例如地震检测。

物理信息神经网络也已被证明在精度测试中优于一阶快速扫描解[175],尤其是在各向异性模型中。

另一种方法涉及使用合成和患者数据进行学习，从电解剖图中获取心脏组织纤维方向，该模型采用各向异性艾克顿方程 [53]。在其实现中，作者在损失函数中添加了针对电导率向量的总变分正则化。

通过忽略各向异性，Sahli Costabal 等人 [151] 也研究了心脏激活映射，其中物理信息神经网络与随机先验函数结合使用，以量化数据不确定性并创建自适应采样策略，用于获取和创建激活图。

横磁极化激励下弱非均匀二维 (2D) 媒体的亥姆霍兹方程在陈等人 [30] 中被提出如下：

$$\nabla^2 E_z(x, y) + \varepsilon_r(x, y)\, k_0^2 E_z = 0,$$

而高频亥姆霍兹方程（频域麦克斯韦方程）则在方和战 [45] 中得到解决。

3.2.2 Unsteady PDEs

Unsteady PDEs usually describe the evolution in time of a physics phenomena. Also in this case, PINNs have proven their reliability in solving such type of problems resulting in a flexible methodology.

3.2.2.1 Advection–Diffusion–Reaction Problems Originally Raissi et al [146] addressed unsteady state problem as:

$$\begin{aligned} u_t &= \mathcal{F}_x(u(x)) \quad x \in \Omega, \\ \mathcal{B}(u(x)) &= 0 \quad x \in \partial\Omega \end{aligned}$$

where $\mathcal{F}_x$ typically contains differential operators of the variable x. In particular a general advection-diffusion reaction equation can be written by setting

$$\mathcal{F}_x(u(x); b, \mu, \sigma) = -div(\mu\nabla u)) + b\nabla u + \sigma u,$$

where, given the parameters b, μ, σ, $-div(\mu\nabla u))$ is the *diffusion* term, while the advection term is $b\nabla u$ which is also known as *transport* term, and finally σu is the *reaction* term.

Diffusion Problems

For a composite material, Amini Niaki et al [6] study a system of equations, that models heat transfer with the known heat equation,

$$\frac{\partial T}{\partial t} = a\frac{\partial^2 T}{\partial x^2} + b\frac{d\alpha}{dt}$$

where a, b are parameters, and a second equation for internal heat generation expressed as a derivative of time of the degree of cure $\alpha \in (0, 1)$ is present.

Amini Niaki et al [6] propose a PINN composed of two disconnected subnetworks and the use of a sequential training algorithm that automatically adapts the weights in the loss, hence increasing the model's prediction accuracy.

Based on physics observations, an activation function with a positive output parameter and a non-zero derivative is selected for the temperature describing network's last layer, i.e. a Softplus activation function, that is a smooth approximation to the ReLU activation function. The Sigmoid function is instead chosen for the last layer of the network that represents the degree of cure. Finally, because of its smoothness and non-zero derivative, the hyperbolic-tangent function is employed as the activation function for all hidden layers.

Since accurate exotherm forecasts are critical in the processing of composite materials inside autoclaves, Amini Niaki et al [6] show that PINN correctly predict the maximum part temperature, i.e. exotherm, that occurs in the center of the composite material due to internal heat.

A more complex problem was addressed in Cai et al [22], where the authors study a kind of free boundary problem, known as the Stefan problem.

The Stefan problems can be divided into two types: the direct Stefan problem, in its traditional form, entails determining the temperature distribution in a domain during a phase transition. The latter, inverse problem, is distinguished by a set of free boundary conditions known as Stefan conditions Wang and Perdikaris [178].

The authors characterize temperature distributions using a PINN model, that consists of a DNN to represent the unknown interface and another FCNN with two outputs, one for each phase. This leads to three residuals, each of which is generated using three neural

3.2.2 非稳态偏微分方程

非稳态偏微分方程通常描述物理现象随时间的演变。在这种情况下，物理信息神经网络（PINNs）已证明其在解决此类问题上的可靠性，从而形成了一种灵活的方法。

3.2.2.1 对流-扩散-反应问题 最初，Raissi等人 [146] 处理了非稳态问题，如下：

$$\begin{aligned} u_t &= \mathcal{F}_x(u(x)) \quad x \in \Omega, \\ \mathcal{B}(u(x)) &= 0 \quad x \in \partial\Omega \end{aligned}$$

其中 $\mathcal{F}_x$ 通常包含关于变量 x 的微分算子。特别是，一个通用的对流-扩散-反应方程可以通过设置

$$\mathcal{F}_x(u(x); b, \mu, \sigma) = -div(\mu\nabla u)) + b\nabla u + \sigma u,$$

在给定参数 b $\mu, \sigma, -div(\mu\nabla u))$ 的情况下，其中 扩散 项是 μ，对流项是 $b\nabla u$，它也被称为 传输 项，最后 σu 是 反应 项。

扩散问题

对于复合材料，Amini Niaki等人 [6] 研究了一个方程组，该方程组使用已知的热方程模拟热传递，

$$\frac{\partial T}{\partial t} = a\frac{\partial^2 T}{\partial x^2} + b\frac{d\alpha}{dt}$$

其中 a,b 是参数，并且存在一个表示固化程度 $\alpha \in (0, 1)$ 对时间导数的内部热生成第二方程。

Amini Niaki等人 [6] 提出了一种由两个不连接的子网络组成的PINN，并使用顺序训练算法自动调整损失中的权重，从而提高模型的预测精度。

基于物理观察，为描述温度的网络的最后一层选择了一个具有正输出参数和非零导数的激活函数，即Softplus激活函数，它是对ReLU激活函数的平滑近似。相反，Sigmoid函数被选择用于表示固化程度的网络的最后一层。最后，由于它的平滑性和非零导数，双曲正切函数被用作所有隐藏层的激活函数。

由于准确的放热预测对于烘箱内复合材料加工至关重要，阿米尼·尼亚基等人 [6] 表明物理信息神经网络（PINN）能够正确预测由于内部热量在复合材料中心产生的最大部件温度，即放热。

Cai等人 [22], 研究了一种更复杂的问题，即自由边界问题，也称为Stefan问题。

Stefan问题可以分为两种类型：传统的直接Stefan问题，其涉及在相变过程中确定域内的温度分布。后者，即逆问题，以一组称为Stefan条件的自由边界条件为特征，王和佩德里卡里斯 [178]。

作者们使用PINN模型表征温度分布，该模型由一个深度神经网络（DNN）表示未知界面，以及另一个具有两个输出的全连接神经网络（FCNN），每个相一个。这导致了三个残差，每个残差都是使用三个神经网络生成的。

networks, namely the two phases $u_\theta^{(1)}, u_\theta^{(2)}$, as well as the interface s_β that takes the boundary conditions into consideration. The two sets of parameters θ and β are minimized through the mean squared errors losses:

$$\mathcal{L}_{\mathcal{F}}(\theta) = \mathcal{L}_r^{(1)}(\theta) + \mathcal{L}_r^{(2)}(\theta)$$

enforces the two PDEs of the heat equation, one for each phase state:

$$\mathcal{L}_r^{(k)}(\theta) = \frac{1}{N_c}\sum_{i=1}^{N_c}\left\|\frac{\partial u_\theta^{(k)}}{\partial t}(x^i, t^i) - \omega_k \frac{\partial^2 u_\theta^{(k)}}{\partial x^2}(x^i, t^i)\right\|^2, \qquad k = 1, 2.$$

on a set of randomly selected collocation locations $\{(x^i, t^i)\}_{i=1}^{N_c}$, and ω_1, ω_2 are two additional training parameters. While, as for the boundary and initial conditions:

$$\mathcal{L}_{\mathcal{B}}(\theta) = \mathcal{L}_{s_{bc}}^{(1)}(\theta, \beta) + \mathcal{L}_{s_{bc}}^{(2)}(\theta, \beta) + \mathcal{L}_{s_{Nc}}(\theta, \beta) + \mathcal{L}_{s_0}(\beta)$$

where $\mathcal{L}_{s_{bc}}^{(k)}$ are the boundary condition of $u^{(k)}$ on the moving boundary $s(t)$, $\mathcal{L}_{s_{Nc}}$ is the free boundary Stefan problem equation, and $\mathcal{L}_{s_0}$ is the initial condition on the free boundary function. Finally, as for the data,

$$\mathcal{L}_{data}(\theta) = \frac{1}{N_d}\sum_{i=1}^{N_d}\|u_\theta(x_{data}^i, t_{data}^i) - u_i^*\|^2,$$

computes the error of the approximation $u(x, t)$ at known data points.

With the previous setup the authors in Cai et al [22] find an accurate solution, however, the basic PINN model fails to appropriately identify the unknown thermal diffusive values, for the inverse problem, due to a local minimum in the training procedure.

So they employ a dynamic weights technique [179], which mitigates problems that arise during the training of PINNs due to stiffness in the gradient flow dynamics. The method significantly minimizes the relative prediction error, showing that the weights in the loss function are crucial and that choosing ideal weight coefficients can increase the performance of PINNs [22].

In Wang and Perdikaris [178], the authors conclude that the PINNs prove to be versatile in approximating complicated functions like the Stefan problem, despite the absence of adequate theoretical analysis like approximation error or numerical stability.

Advection Problems

In He and Tartakovsky [59], multiple advection-dispersion equations are addressed, like

$$u_t + \nabla\cdot(-\kappa\nabla u + \mathbf{v}u) = s$$

where κ is the dispersion coefficient.

The authors find that the PINN method is accurate and produces results that are superior to those obtained using typical discretization-based methods. Moreover both Dwivedi and Srinivasan [41] and He and Tartakovsky [59] solve the same 2D advection-dispersion equation,

$$u_t + \nabla\cdot(-\kappa\nabla u + \mathbf{a}u) = 0$$

In this comparison, the PINN technique [59] performs better that the ELM method [41], given the errors that emerge along borders, probably due to larger wights assigned to boundary and initial conditions in He and Tartakovsky [59].

网络，即两个相$u_\theta^{(1)}$、$u_\theta^{(2)}$，以及考虑边界条件的界面s_β。两组参数θ和β通过均方误差损失最小化：

$$\mathcal{L}_{\mathcal{F}}(\theta) = \mathcal{L}_r^{(1)}(\theta) + \mathcal{L}_r^{(2)}(\theta)$$

强制执行热方程的两个偏微分方程，每个相态一个：

$$\mathcal{L}_r^{(k)}(\theta) = \frac{1}{N_c}\sum_{i=1}^{N_c}\left\|\frac{\partial u_\theta^{(k)}}{\partial t}(x^i, t^i) - \omega_k \frac{\partial^2 u_\theta^{(k)}}{\partial x^2}(x^i, t^i)\right\|^2, \qquad k = 1, 2.$$

在随机选择的配置位置集合$\{(x^i,\ \ t^i)\}^{N_c}{}_{i=1}$，以及$\omega_1, \omega_2$是两个额外的训练参数。而，对于边界和初始条件：

$$\mathcal{L}_{\mathcal{B}}(\theta) = \mathcal{L}_{s_{bc}}^{(1)}(\theta, \beta) + \mathcal{L}_{s_{bc}}^{(2)}(\theta, \beta) + \mathcal{L}_{s_{Nc}}(\theta, \beta) + \mathcal{L}_{s_0}(\beta)$$

其中$\mathcal{L}^{(k)}{}_{s_{bc}}$是移动边界$s(t)$上$u^{(k)}$的边界条件，$\mathcal{L}_{s_{Nc}}$是自由边界斯蒂芬问题方程，而$\mathcal{L}_{s0}$是自由边界函数上的初始条件。最后，关于数据，

$$\mathcal{L}_{data}(\theta) = \frac{1}{N_d}\sum_{i=1}^{N_d}\|u_\theta(x_{data}^i, t_{data}^i) - u_i^*\|^2,$$

在已知数据点上计算逼近误差$u(x,t)$。

在之前的设置下，蔡等人 [22] 找到了一个精确的解，然而，由于训练程序中存在局部最小值，基本的物理信息神经网络模型无法适当识别未知的热扩散值，对于逆问题。

因此，他们采用了一种动态权重技术 [179],，该技术减轻了由于梯度流动力学的刚度而在物理信息神经网络训练过程中出现的问题。该方法显著降低了相对预测误差，表明损失函数中的权重至关重要，并且选择理想的权重系数可以提高物理信息神经网络的性能 [22]。

在王和佩德里卡里斯 [178]，中，作者得出结论，尽管缺乏足够的理论分析，如近似误差或数值稳定性，物理信息神经网络被证明在逼近复杂函数（如Stefan问题）方面具有多功能性。

平流问题

在何和塔塔科夫斯基 [59]，中，解决了多个平流弥散方程，例如

$$u_t + \nabla\cdot(-\kappa\nabla u + \mathbf{v}u) = s$$

其中 κ 是弥散系数。

作者发现，PINN方法准确，其结果优于使用典型离散化方法得到的结果。此外，Dwivedi和Srinivasan [41] 以及 He和Tartakovsky [59] 都求解同一个2D对流-扩散方程，

$$u_t + \nabla\cdot(-\kappa\nabla u + \mathbf{a}u) = 0$$

在这项比较中，PINN技术 [59] 比ELM方法 [41]，表现更好，这是由于边界处出现的误差，可能是因为He和Tartakovsky [59] 给边界和初始条件分配了更大的权重。

Moreover, in Dwivedi and Srinivasan [41], an interesting case of PINN and PIELM failure in solving the linear advection equation is shown, involving PDE with sharp gradient solutions.

He et al [60] solve Darcy and advection-dispersion equations proposing a Multiphysics-informed neural network (MPINN) for subsurface transport problems, and also explore the influence of the neural network size on the accuracy of parameter and state estimates.

In Schiassi et al [153], a comparison of two methods is shown, Deep–TFC and X–TFC, on how the former performs better in terms of accuracy when the problem becomes sufficiently stiff. The examples are mainly based on 1D time-dependent Burgers' equation and the Navier–Stokes (NS) equations.

In the example of the two-dimensional Burgers equation, Jagtap et al [71] demonstrate that by having an approximate a priori knowledge of the position of shock, one can appropriately partition the domain to capture the steep descents in solution. This is accomplished through the cPINN domain decomposition flexibility.

While Arnold and King [9] addressed the Burgers equations in the context of model predictive control (MPC).

In Meng et al [109] the authors study a two-dimensional diffusion-reaction equation that involves long-time integration and they use a parareal PINN (PPINN) to divide the time interval into equal-length sub-domains. PPINN is composed of a fast coarse-grained (CG) solver and a finer solver given by PINN.

3.2.2.2 Flow Problems

Particular cases for unsteady differential problems are the ones connected to the motion of fluids. Navier–Stokes equations are widely present in literature, and connected to a large number of problems and disciplines. This outlines the importance that reliable strategies for solving them has for the scientific community. Many numerical strategies have been developed to solve this kind of problems. However, computational issues connected to specific methods, as well as difficulties that arise in the choice of the discrete spatio-temporal domain may affect the quality of numerical solution. PINNs, providing mesh-free solvers, may allow to overcome some issues of standard numerical methods, offering a novel perspective in this field.

Navier–Stokes Equations

Generally Navier–Stokes equations are written as

$$\mathcal{F}_x(u(x);\nu,p) = -div[\nu(\nabla u + \nabla u^T)] + (u+\nabla)u + \nabla p - f,$$

where, u is the speed of the fluid, p the pressure and ν the viscosity [136]. The dynamic equation is coupled with

$$div(u) = 0$$

for expressing mass conservation.

The Burgers equation, a special case of the Navier–Stokes equations, was covered in the previous section.

Using quick parameter sweeps, Arthurs and King [10] demonstrate how PINNs may be utilized to determine the degree of narrowing in a tube. PINNs are trained using finite element data to estimate Navier–Stokes pressure and velocity fields throughout a parametric domain. The authors present an active learning algorithm (ALA) for training PINNs to predict PDE solutions over vast areas of parameter space by combining ALA, a domain and mesh generator, and a traditional PDE solver with PINN.

此外，在Dwivedi和Srinivasan [41], 中展示了一个有趣的PINN和PIELM在求解线性平流方程时失败的案例，涉及具有陡峭梯度解的偏微分方程。

何等 [60] 求解达西方程和对流弥散方程，提出了一种多物理场信息神经网络（MPINN）用于地下输运问题，并探讨了神经网络大小对参数和状态估计精度的影响。

在Schiassi等人 [153], 中展示了对两种方法的比较，深度-TFC和X-TFC，在问题变得足够刚化时，前者在精度方面表现更好。示例主要基于1D时变伯格斯方程和纳维-斯托克斯（NS）方程。

在二维伯格斯方程的示例中，Jagtap等人 [71] 证明了通过近似先验知识了解冲击波的位置，可以适当划分域以捕捉解中的陡峭下降。这是通过cPINN域分解灵活性实现的。

阿诺德和King [9] 在模型预测控制（MPC）的背景下研究了布格斯方程。

在Meng等人 [109] 中，作者研究了一个涉及长时间积分的二阶扩散-反应方程，并使用并行物理信息神经网络（PPINN）将时间区间划分为等长的子域。PPINN由一个快速粗粒度（CG）求解器和一个由物理信息神经网络给出的更精细的求解器组成。

3.2.2.2 流体问题

非稳态微分问题的特例是与流体运动相关的问题。纳维-斯托克斯方程在文献中广泛存在，并与大量问题和学科相关联。这突出了为解决它们开发可靠策略对科学界的重要性。许多数值策略已被开发用于解决这类问题。然而，与特定方法相关的计算问题，以及在选择离散时空域时出现的问题，可能会影响数值解的质量。物理信息神经网络提供无网格求解器，可能允许克服标准数值方法的一些问题，为该领域提供一种新的视角。

纳维-斯托克斯方程

通常纳维-斯托克斯方程表示为

$$\mathcal{F}_x(u(x);\nu,p) = -div[\nu(\nabla u + \nabla u^T)] + (u+\nabla)u + \nabla p - f,$$

其中，u 是流体的速度，p 是压力，ν 是粘度 [136]。动态方程与

$$div(u) = 0$$

用于表达质量守恒。

布格斯方程是纳维-斯托克斯方程的一个特例，在上一节中已介绍。

使用快速参数扫描，Arthurs和King [10] 展示了如何使用物理信息神经网络（PINNs）来确定管道的收缩程度。PINNs使用有限元数据来估计参数域内纳维-斯托克斯压力和速度场。作者提出了一种主动学习算法（ALA），用于训练PINNs，通过结合ALA、域和网格生成器以及传统偏微分方程（PDE）求解器与PINN，来预测参数空间广阔区域内的PDE解。

PINNs are also applied on the drift-reduced Braginskii model by learning turbulent fields using limited electron pressure data [106]. The authors simulated synthetic plasma using the global drift-ballooning (GDB) finite-difference algorithm by solving a fluid model, ie. two-fluid drift-reduced Braginskii equations. They also observe the possibility to infer 3D turbulent fields from only 2D observations and representations of the evolution equations. This can be used for fluctuations that are difficult to monitor or when plasma diagnostics are unavailable.

Xiao et al [186] review available turbulent flow databases and propose benchmark datasets by systematically altering flow conditions.

Zhu et al [199] predict the temperature and melt pool fluid dynamics in 3D metal additive manufacturing AM processes.

The thermal-fluid model is characterized by Navier–Stokes equations (momentum and mass conservation), and energy conservation equations.

They approach the Dirichlet BC in a "hard" manner, employing a specific piece of the neural network to solely meet the prescribed Dirichlet BC; while Neumann BCs, that account for surface tension, are treated conventionally by adding the term to the loss function. They choose the loss weights based on the ratios of the distinct components of the loss function [199].

Cheng and Zhang [31] solve fluid flows dynamics with Res–PINN, PINN paired with a Resnet blocks, that is used to improve the stability of the neural network. They validate the model with Burgers' equation and Navier–Stokes (N–S) equation, in particular, they deal with the cavity flow and flow past cylinder problems. A curious phenomena observed by Cheng and Zhang [31] is a difference in magnitude between the predicted and actual pressure despite the fact that the distribution of the pressure filed is essentially the same.

To estimate the solutions of parametric Navier–Stokes equations, Sun et al [168] created a physics-constrained, data-free, FC–NN for incompressible flows. The DNN is trained purely by reducing the residuals of the governing N–S conservation equations, without employing CFD simulated data. The boundary conditions are also hard-coded into the DNN architecture, since the aforementioned authors claim that in data-free settings, "hard" boundary enforcement is preferable than "soft" boundary approach.

Three flow examples relevant to cardiovascular applications were used to evaluate the suggested approaches.

In particular, the Navier–Stokes equations are given [168] as:

$$\mathcal{F}(\mathbf{u}, p) = 0 := \begin{cases} \nabla \cdot \mathbf{u} = 0, & \mathbf{x}, t \in \Omega, \boldsymbol{\gamma} \in \mathbb{R}^d, \\ \dfrac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} + \dfrac{1}{\rho}\nabla p - \nu \nabla^2 \mathbf{u} + \mathbf{b}_f = 0, & \mathbf{x}, t \in \Omega, \boldsymbol{\gamma} \in \mathbb{R}^d \end{cases}$$

where $\boldsymbol{\gamma}$ is a parameter vector, and with

$$\begin{aligned} \mathcal{I}(\mathbf{x}, p, \mathbf{u}, \boldsymbol{\gamma}) &= 0, \qquad \mathbf{x} \in \Omega, t = 0, \boldsymbol{\gamma} \in \mathbb{R}^d, \\ \mathcal{B}(t, \mathbf{x}, p, \mathbf{u}, \boldsymbol{\gamma}) &= 0, \qquad \mathbf{x}, t \in \partial\Omega \times [0, T], \boldsymbol{\gamma} \in \mathbb{R}^d, \end{aligned}$$

where $\mathcal{I}$ and $\mathcal{B}$ are generic differential operators that determine the initial and boundary conditions.

The boundary conditions (IC/BC) are addressed individually in Sun et al [168]. The Neumann BC are formulated into the equation loss, i.e., in a soft manner, whereas the IC and Dirichlet BC are encoded in the DNN, i.e., in a hard manner.

物理信息神经网络 (PINNs) 也被应用于漂移减少的布拉金斯基模型，通过学习有限电子压力数据 [106] 来获取湍流场。作者使用全局漂移气球 (GDB) 有限差分算法，通过求解流体模型（即两流体漂移减少布拉金斯基方程）模拟了合成等离子体。他们还观察到从 2D 观测和演化方程的表示中推断 3D 湍流场的可能性。这可用于难以监测的波动或等离子体诊断不可用时的情况。

萧等人 [186] 综述了现有的湍流数据库，并通过系统地改变流动条件提出了基准数据集。

朱等人 [199] 预测了 3D 金属增材制造 AM 过程中的温度和熔池流体动力学。

热流体模型的特点是纳维-斯托克斯方程（动量和质量守恒）以及能量守恒方程。

他们以“硬”的方式处理狄利克雷边界条件，利用神经网络的一个特定部分来仅满足规定的狄利克雷边界条件；而诺伊曼边界条件，即考虑表面张力的边界条件，则通过将项添加到损失函数中采用传统方法进行处理。他们根据损失函数的不同组成部分的比率选择损失权重[199]。

程和张 [31] 使用Res-PINN（Resnet块与PINN结合）解决流体流动力学问题，该Resnet块用于提高神经网络的稳定性。他们使用伯格斯方程和纳维-斯托克斯(N-S)方程验证模型，特别是他们处理了空腔流动和绕圆柱流动问题。程和张 [31] 观察到的一个有趣现象是，尽管压力场的分布基本上是相同的，但预测压力和实际压力之间存在幅值差异。

为了估计参数化纳维-斯托克斯方程的解，孙等人 [168] 为不可压缩流动创建了一个物理约束、无数据、全连接神经网络(FC-NN)。深度神经网络(DNN)纯粹通过减少控制性N-S守恒方程的残差进行训练，而没有使用CFD模拟数据。边界条件也被硬编码到DNN架构中，因为上述作者声称，在无数据设置中，“硬”边界执行比“软”边界方法更可取。

使用了与心血管应用相关的三个流动示例来评估所建议的方法。

特别是，纳维-斯托克斯方程给出 [168]为:

$$\mathcal{F}(\mathbf{u}, p) = 0 := \begin{cases} \nabla \cdot \mathbf{u} = 0, & \mathbf{x}, t \in \Omega, \boldsymbol{\gamma} \in \mathbb{R}^d, \\ \dfrac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} + \dfrac{1}{\rho}\nabla p - \nu \nabla^2 \mathbf{u} + \mathbf{b}_f = 0, & \mathbf{x}, t \in \Omega, \boldsymbol{\gamma} \in \mathbb{R}^d \end{cases}$$

其中 $\boldsymbol{\gamma}$ 是一个参数向量，并且有

$$\begin{aligned} \mathcal{I}(\mathbf{x}, p, \mathbf{u}, \boldsymbol{\gamma}) &= 0, \qquad \mathbf{x} \in \Omega, t = 0, \boldsymbol{\gamma} \in \mathbb{R}^d, \\ \mathcal{B}(t, \mathbf{x}, p, \mathbf{u}, \boldsymbol{\gamma}) &= 0, \qquad \mathbf{x}, t \in \partial\Omega \times [0, T], \boldsymbol{\gamma} \in \mathbb{R}^d, \end{aligned}$$

其中 $\mathcal{I}$ 和 $\mathcal{B}$ 是通用微分算子，它们决定了初始和边界条件。

边界条件（初始/边界条件）在孙等人 [168]中分别处理。诺伊曼边界条件被公式化为方程损失，即以软方式，而初始条件和狄利克雷边界条件被编码在深度神经网络中，即以硬方式。

As a last example, NSFnets [73] has been developed considering two alternative mathematical representations of the Navier–Stokes equations: the velocity-pressure (VP) formulation and the vorticity-velocity (VV) formulation.

Hyperbolic Equations

Hyperbolic conservation law is used to simplify the Navier–Stokes equations in hemodynamics [81].

Hyperbolic partial differential equations are also addressed by Abreu and Florindo [1]: in particular, they study the inviscid nonlinear Burgers' equation and 1D Buckley–Leverett two-phase problem. They actually try to address problems of the following type:

$$\frac{\partial u}{\partial t} + \frac{\partial H(u)}{\partial x} = 0, \quad x \in \mathbb{R}, \quad t > 0, \qquad u(x,0) = u_0(x),$$

whose results were compared with those obtained by the Lagrangian–Eulerian and Lax–Friedrichs schemes. While Patel et al [130] proposes a PINN for discovering thermodynamically consistent equations that ensure hyperbolicity for inverse problems in shock hydrodynamics.

Euler equations are hyperbolic conservation laws that might permit discontinuous solutions such as shock and contact waves, and in particular a one dimensional Euler system is written as [71]

$$\frac{\partial U}{\partial t} + \nabla \cdot f(U) = 0, \ x \in \Omega \subset \mathbb{R}^2,$$

where

$$U = \begin{bmatrix} \rho \\ \rho u \\ \rho E \end{bmatrix} \qquad f = \begin{bmatrix} \rho u, \\ p + \rho u^2 \\ pu + \rho u E \end{bmatrix}$$

given ρ as the density, p as the pressure, u the velocity, and E the total energy. These equations regulate a variety of high-speed fluid flows, including transonic, supersonic, and hypersonic flows. Mao et al [103] can precisely capture such discontinuous flows solutions for one-dimensional Euler equations, which is a challenging task for existing numerical techniques. According to Mao et al [103], appropriate clustering of training data points around a high gradient area can improve solution accuracy in that area and reduces error propagation to the entire domain. This enhancement suggests the use of a separate localized powerful network in the region with high gradient solution, resulting in the development of a collection of individual local PINNs with varied features that comply with the known prior knowledge of the solution in each sub-domain. As done in Jagtap et al [71], cPINN splits the domain into a number of small subdomains in which multiple neural networks with different architectures (known as sub-PINN networks) can be used to solve the same underlying PDE.

Still in reference to Mao et al [103], the authors solve the one-dimensional Euler equations and a two-dimensional oblique shock wave problem. The authors can capture the solutions with only a few scattered points distributed randomly around the discontinuities. The above-mentioned paper employ density gradient and pressure $p(x,t)$ data, as well as conservation laws, to infer all states of interest (density, velocity, and pressure fields) for the inverse problem without employing any IC/BCs. They were inspired by the experimental photography technique of Schlieren.

作为最后一个例子，NSFnets [73] 已被开发，考虑了纳维-斯托克斯方程的两个替代数学表示：速度-压力（VP）格式和涡量-速度（VV）格式。

双曲型方程

双曲守恒律用于简化血流动力学 [81] 中的纳维-斯托克斯方程。

双曲偏微分方程也由 Abreu 和 Florindo [1] 讨论：特别是，他们研究了无粘非线性布热斯方程和 1D 巴克莱-勒维特两相问题。他们实际上试图解决以下类型的问题：

$$\frac{\partial u}{\partial t} + \frac{\partial H(u)}{\partial x} = 0, \quad x \in \mathbb{R}, \quad t > 0, \qquad u(x,0) = u_0(x),$$

其结果与拉格朗日-欧拉和Lax-Friedrichs方案获得的结果进行了比较。而Patel等人 [130] 提出了一种PINN，用于发现确保冲击水动力学逆问题中双曲性的热力学一致性方程。

欧拉方程是双曲守恒律，可能允许不连续解，如冲击波和接触波，特别是一维欧拉系统写为 [71]。

$$\frac{\partial U}{\partial t} + \nabla \cdot f(U) = 0, \ x \in \Omega \subset \mathbb{R}^2,$$

其中

$$U = \begin{bmatrix} \rho \\ \rho u \\ \rho E \end{bmatrix} \qquad f = \begin{bmatrix} \rho u, \\ p + \rho u^2 \\ pu + \rho u E \end{bmatrix}$$

给定 ρ 作为密度，p 作为压力，u 作为速度，以及E 作为总能量。这些方程调节了各种高速流体流动，包括跨音速、超音速和高超音速流动。毛等人 [103] 能够精确捕捉一维欧拉方程的不连续流动解，这对现有的数值技术来说是一项具有挑战性的任务。根据毛等人 [103], ，在梯度区域周围适当聚类训练数据点可以提高该区域的解精度，并减少误差传播到整个域。这种改进表明，在具有高梯度解的区域使用一个单独的局部强大网络，导致开发了一系列具有不同特征的独立局部PINNs，这些特征符合每个子域中解的已知先验知识。正如Jagtap等人 [71], 所做的那样，cPINN将域划分为许多小子域，在这些子域中可以使用具有不同架构（称为子PINN网络）的多个神经网络来求解相同的底层偏微分方程。

仍然参考毛等人 [103], ，作者求解了一维欧拉方程和二维斜冲击波问题。作者仅用少量随机分布在间断点周围的散点即可捕捉解。上述论文采用密度梯度和压力 $p(x,t)$ 数据，以及守恒律，来推断逆问题中所有感兴趣的状态（密度、速度和压力场），而无需使用任何初始/边界条件。他们受到了阴影成像技术的实验摄影技术的启发。

They also illustrate that the position of the training points is essential for the training process. The results produced by combining the data with the Euler equations in characteristic form outperform the results obtained using conservative forms.

3.2.2.3 Quantum Problems A 1D nonlinear Schrödinger equation is addressed in Raissi [139], and Raissi et al [146] as:

$$i\frac{\partial \psi}{\partial t}+\frac{1}{2}\frac{\partial^2 \psi}{\partial x^2}+|\psi|^2\psi=0 \tag{10}$$

where $\psi(x,t)$ is the complex-valued solution. This problem was chosen to demonstrate the PINN's capacity to handle complex-valued answers and we will develop this example in Sect. 3.4. Also a quantum harmonic oscillator (QHO)

$$i\frac{\partial \psi(x,t)}{\partial t}+\frac{1}{2}\Delta\psi(x,t)-V(x,t)=0$$

is addressed in Stiller et al [167], with V a scalar potential. They propose a gating network that determines which MLP to use, while each MLP consists of linear layers and tanh activation functions; so the solution becomes a weighted sum of MLP predictions. The quality of the approximated solution of PINNs was comparable to that of state-of-the-art spectral solvers that exploit domain knowledge by solving the equation in the Fourier domain or by employing Hermite polynomials.

Vector solitons, which are solitary waves with multiple components in the the coupled nonlinear Schrödinger equation (CNLSE) is addressed by Mo et al [115], who extended PINNs with a pre-fixed multi-stage training algorithm. These findings can be extendable to similar type of equations, such as equations with a rogue wave [176] solution or the Sasa–Satsuma equation and Camassa–Holm equation.

3.3 Other Problems

Physics Informed Neural Networks have been also applied to a wide variety of problems that go beyond classical differential problems. As examples, in the following, the application of such a strategy as been discussed regarding Fractional PDEs and Uncertainty estimation.

3.3.1 Differential Equations of Fractional Order

Fractional PDEs can be used to model a wide variety of phenomenological properties found in nature with parameters that must be calculated from experiment data, however in the spatiotemporal domain, field or experimental measurements are typically scarce and may be affected by noise [128]. Because automatic differentiation is not applicable to fractional operators, the construction of PINNs for fractional models is more complex. One possible solution is to calculate the fractional derivative using the L1 scheme [108].

In the first example from Mehta et al [108] they solve a turbulent flow with one dimensional mean flow.

In Pang et al [128], the authors focus on identifying the parameters of fractional PDEs with known overall form but unknown coefficients and unknown operators, by giving rise to fPINN. They construct the loss function using a hybrid technique that includes both automatic differentiation for integer-order operators and numerical discretization for fractional operators. They also analyse the convergence of fractional advection-diffusion equations (fractional ADEs)

他们还表明，训练点的位置对于训练过程至关重要。将数据与特征形式的欧拉方程相结合所产生的结果优于使用守恒形式获得的结果。

3.2.2.3 量子问题在Raissi[139], 和Raissi等人 [146] 中，研究了1维非线性薛定谔方程：

$$i\frac{\partial \psi}{\partial t}+\frac{1}{2}\frac{\partial^2 \psi}{\partial x^2}+|\psi|^2\psi=0 \tag{10}$$

其中 $\psi(x,t)$ 是复值解。这个问题被选择来展示PINN处理复值解的能力，我们将在第 3.4节中开发这个示例。此外，量子谐振子（QHO）

$$i\frac{\partial \psi(x,t)}{\partial t}+\frac{1}{2}\Delta\psi(x,t)-V(x,t)=0$$

在Stiller等人 [167],中，使用 V标量势进行了研究。他们提出了一种门控网络，该网络确定使用哪个多层感知机，而每个多层感知机由线性层和tanh激活函数组成；因此，解成为多层感知机预测的加权总和。PINNs的近似解质量与利用域知识通过在傅里叶域中求解方程或采用埃尔米特多项式而实现的最先进谱求解器相当。

矢量孤子，即耦合非线性薛定谔方程（CNLSE）中的多组分孤立波，由Mo等人 [115], 通过引入预置的多阶段训练算法扩展了物理信息神经网络（PINNs）来研究。这些发现可推广到类似类型的方程，例如具有rogue波 [176] 解的方程或Sasa–Satsuma方程和Camassa–Holm方程。

3.3 其他问题

物理信息神经网络已被应用于超越经典微分问题的一系列广泛问题。作为示例，在以下内容中，我们将讨论将此类策略应用于分数阶偏微分方程（Fractional PDEs）和不确定性估计的应用。

3.3.1 Differential Equations of Fractional Order

分数阶偏微分方程可用于模拟自然界中广泛存在的现象学属性，其参数必须从实验数据中计算得出，然而在时空域中，场或实验测量通常稀缺且可能受噪声 [128]影响。由于自动微分不适用于分数阶算子，分数模型的物理信息神经网络的构建更为复杂。一种可能的解决方案是使用L1方案 [108]计算分数阶导数。

在Mehta等人 [108] 的第一个例子中，他们求解了一个具有一维平均流动的湍流。

在Pang等人 [128],中，作者们专注于识别具有已知整体形式但未知系数和未知算子的分数阶偏微分方程的参数，通过引入fPINN。他们使用一种混合技术构建损失函数，该技术包括对整数阶算子的自动微分和对分数阶算子的数值离散化。他们还分析了分数阶平流扩散方程（分数阶ADEs）的收敛性

The solution proposed in Pang et al [128], is then extended in Kharazmi et al [77] where they address also time-dependent fractional orders. The formulation in Kharazmi et al [77] uses separate neural network to represent each fractional order and use a large neural network to represent states.

3.3.2 Uncertainty Estimation

In data-driven PDE solvers, there are several causes of uncertainty. The quality of the training data has a significant impact on the solution's accuracy.

To address forward and inverse nonlinear problems represented by partial differential equations (PDEs) with noisy data, Yang et al [190] propose a Bayesian physics-informed neural network (B-PINN). The Bayesian neural network acts as the prior in this Bayesian framework, while an Hamiltonian Monte Carlo (HMC) method or variational inference (VI) method can be used to estimate the posterior.

B-PINNs [190] leverage both physical principles and scattered noisy observations to give predictions and quantify the aleatoric uncertainty coming from the noisy data.

Yang et al [190] test their network on some forward problems (1D Poisson equation, Flow in one dimension across porous material with a boundary layer, Nonlinear Poisson equation in one dimension and the 2D Allen–Cahn equation), while for inverse problems the 1D diffusion-reaction system with nonlinear source term and 2D nonlinear diffusion-reaction system are addressed.

Yang et al [190] use also the B-PINNs for a high-dimensional diffusion-reaction system, where the locations of three contaminating sources are inferred from a set of noisy data.

Yang et al [189] considers the solution of elliptic stochastic differential equations (SDEs) that required approximations of three stochastic processes: the solution $u(x;\gamma)$, the forcing term $f(x;\gamma)$, and the diffusion coefficient $k(x;\gamma)$.

In particular, Yang et al [189] investigates the following, time independent, SDE

$$\begin{aligned}\mathcal{F}_{\mathbf{x}}[u(\mathbf{x};\gamma);k(\mathbf{x};\gamma)] &= f(\mathbf{x};\gamma),\\ B_{\mathbf{x}}[u(\mathbf{x};\gamma)] &= b(\mathbf{x};\gamma),\end{aligned}$$

where $k(x;\gamma)$ and $f(x;\gamma)$ are independent stochastic processes, with k strictly positive.

Furthermore, they investigate what happens when there are a limited number of measurements from scattered sensors for the stochastic processes. They show how the problem gradually transform from forward to mixed, and finally to inverse problem. This is accomplished by assuming that there are a sufficient number of measurements from some sensors for $f(x;\gamma)$, and then as the number of sensors measurements for $k(x;\gamma)$ is decreased, the number of sensors measurements on $u(x;\gamma)$ is increased, and thus a forward problem is obtained when there are only sensors for k and not for u, while an inverse problem has to be solved when there are only sensors for u and not for k.

A similar result was previously obtained by [195], where they used stochastic data from sparse sensors, and a PINN to learn the modal functions of the arbitrary polynomial chaos (aPC) expansion of its solution.

Furthermore, [196] propose two PINNs for solving time-dependent stochastic partial differential equation (SPDE), based on spectral dynamically orthogonal (DO) and borthogonal (BO) stochastic process representation approaches. They tested their PINN on a linear stochastic advection problem, a stochastic Burgers' equation, and a nonlinear reaction-diffusion equation.

Pang等人 [128]，提出的解法随后在哈拉兹米等人 [77] 中得到扩展，其中他们还处理了时变分数阶。哈拉兹米等人 [77] 中的公式使用单独的神经网络来表示每个分数阶，并使用一个大型神经网络来表示状态。

3.3.2 不确定性估计

在数据驱动PDE求解器中，存在多种不确定性来源。训练数据的质量对解的精度有显著影响。

为了解决由偏微分方程 (PDEs) 表示的正向和反向非线性问题，并处理噪声数据，Yang等人 [190] 提出了一种贝叶斯物理信息神经网络（B-PINN）。贝叶斯神经网络在此贝叶斯框架中充当先验，而哈密顿蒙特卡洛（HMC）方法或变分推理（VI）方法可用于估计后验。

B-PINNs [190] 利用物理原理和散乱噪声观测来给出预测并量化来自噪声数据的偶然不确定性。

Yang et al [190] 在（一维泊松方程、一维多孔材料中的流动（带边界层）、一维非线性泊松方程和二维艾伦-卡恩方程）等一些正向问题上测试他们的网络，而在逆向问题上则处理了一维非线性源项扩散-反应系统和二维非线性扩散-反应系统。

Yang et al [190] 也使用B-PINNs来处理一个高维扩散-反应系统，其中三个污染源的位置是从一组噪声数据中推断出来的。

Yang et al [189] 考虑了椭圆随机微分方程（SDEs）的解，该解需要近似三个随机过程：解 $u(x;\gamma)$、强迫项 $f(x;\gamma)$ 和扩散系数 $k(x;\gamma)$。特别是，

Yang 等人 [189] 研究了以下内容，时间无关 end实体，随机微分方程

$$\begin{aligned}\mathcal{F}_{\mathbf{x}}[u(\mathbf{x};\gamma);k(\mathbf{x};\gamma)] &= f(\mathbf{x};\gamma),\\ B_{\mathbf{x}}[u(\mathbf{x};\gamma)] &= b(\mathbf{x};\gamma),\end{aligned}$$

在 $k(x;\gamma)$ 和 $f(x;\gamma)$ 是独立的随机过程，其中 k 严格为正。

此外，他们研究了当随机过程只有散射传感器有限测量时会发生什么。他们展示了问题如何逐渐从正向问题转变为混合问题，最终转变为逆问题。这是通过假设对于 $f(x;\gamma)$ 存在足够数量的传感器测量，然后随着传感器测量 $k(x;\gamma)$ 的数量减少，传感器测量 $u(x;\gamma)$ 的数量增加，从而在只有 k 传感器而没有 u 传感器时得到正向问题，而在只有 u 传感器而没有 k 传感器时必须求解逆问题。

之前已有类似结果，其中他们使用来自稀疏传感器的随机数据，并采用物理信息神经网络(PINN)来学习其解的任意多项式混沌(aPC)展开的模态函数。

此外，[196]基于谱动态正交（DO）和正交（BO）随机过程表示方法，提出了两种用于求解时变随机偏微分方程（SPDE）的物理信息神经网络（PINN）。他们测试了他们的PINN在一个线性随机平流问题、一个随机博克斯-马斯顿方程和一个非线性反应-扩散方程上。

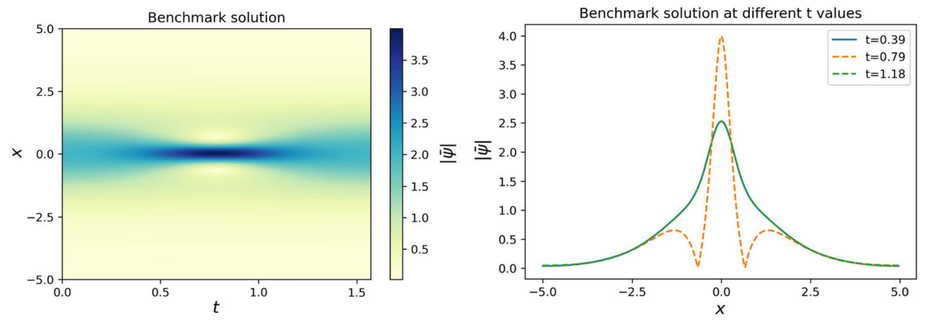

Fig. 3 1D nonlinear Shrödinger (NLS) solution module $|\bar{\psi}|$. The solution is generated with Chebfun open-source software [40] and used as a reference solution. We also show the solution at three different time frames, $t = \frac{\pi}{8}, \frac{\pi}{4}, \frac{3\pi}{8}$. We expect the solution to be symmetric with respect to $\frac{\pi}{4}$

In order to characterizing shape changes (morphodynamics) for cell-drug interactions, Cavanagh et al [27] use kernel density estimation (KDE) for translating morphspace embeddings into probability density functions (PDFs). Then they use a top-down Fokker–Planck model of diffusive development over Waddington-type landscapes, with a PINN learning such landscapes by fitting the PDFs to the Fokker–Planck equation. The architecture includes a neural network for each condition to learn: the PDF, diffusivity, and landscape. All parameters are fitted using approximate Bayesian computing with sequential Monte Carlo (aBc-SMC) methods: in this case, aBc selects parameters from a prior distribution and runs simulations; if the simulations match the data within a certain level of similarity, the parameters are saved. So the posterior distribution is formed by the density over the stored parameters [27].

3.4 Solving a Differential Problem with PINN

Finally, this subsection discusses a realistic example of a 1D nonlinear Shrödinger (NLS) problem, as seen in Fig. 3. The nonlinear problem is the same presented in Raissi [139], Raissi et al [143], used to demonstrate the PINN's ability to deal with periodic boundary conditions and complex-valued solutions.

Starting from an initial state $\psi(x, 0) = 2\,\text{sech}(x)$ and assuming periodic boundary conditions Eq. (10) with all boundary conditions results in the initial boundary value problem, given a domain $\Omega = [-5, 5] \times (0, T]$ written as:

$$\begin{cases} i\psi_t + 0.5\psi_{xx} + |\psi|^2\psi = 0 & (x,t) \in \Omega \\ \psi(0,x) = 2\,\text{sech}(x) & x \in [-5,5] \\ \psi(t,-5) = \psi(t,5) & t \in (0,T] \\ \psi_x(t,-5) = \psi_x(t,5) & t \in (0,T] \end{cases} \tag{11}$$

where $T = \pi/2$.

To assess the PINN's accuracy, Raissi et al [143] created a high-resolution data set by simulating the Schrödinger equation using conventional spectral methods. The authors integrated the Schrödinger equation up to a final time $T = \pi/2$ using the MATLAB based Chebfun open-source software [40].

The PINN solutions are trained on a subset of measurements, which includes initial data, boundary data, and collocation points inside the domain. The initial time data, $t = 0$, are

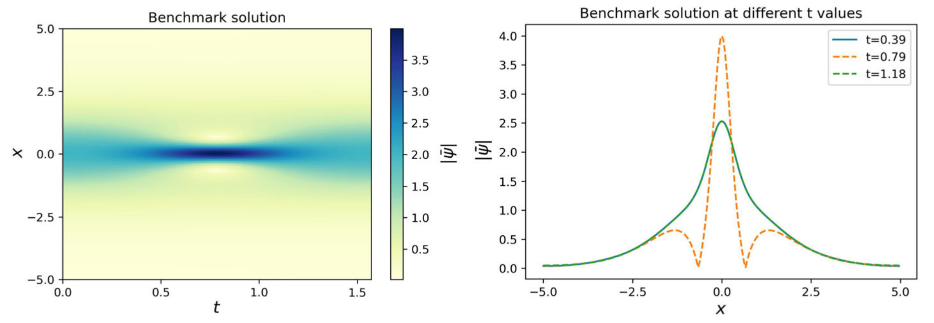

图3 一维非线性薛定谔（NLS）解模块 $|\bar{\psi}|$. 该解使用Chebfun开源软件生成，并用作参考解。我们还展示了在三个不同时间框架 $t = \frac{\pi}{8}, \frac{\pi}{4}, \frac{3\pi}{8}$的解。我们预期该解关于 $\frac{\pi}{4}$

为了表征细胞-药物相互作用的形状变化（形态动力学），Cavanagh等人 [27] 使用核密度估计（KDE）将形态空间嵌入转换为概率密度函数（PDFs）。然后他们使用在瓦丁顿型景观上的扩散发展自上而下福克-普朗克模型，该模型通过将PDFs拟合到福克-普朗克方程来学习景观。该架构包括每个条件的一个神经网络，以学习：PDFs、扩散率和景观。所有参数都使用近似贝叶斯计算与序贯蒙特卡洛（aBc-SMC）方法拟合：在这种情况下，aBc从先验分布中选择参数并运行模拟；如果模拟在相似度水平内与数据匹配，则保存参数。因此，后验分布由存储参数上的密度形成 [27].

3.4 使用PINN求解微分问题

最后，本小节讨论了一个1D非线性薛定谔(NLS)问题的实际例子，如图所示3。这个非线性问题是Raissi [139],Raissi等人 [143], 用来展示PINN处理周期性边界条件和复值解的能力。

从初始状态 $\psi(x\ \ 0) = 2\,\text{sech}(x)$ 出发，并假设周期性边界条件等式(10)以及所有边界条件，得到初始边值问题，给定一个域 $\Omega = [-5, 5] \times (0, T]$，表示为：

$$\begin{cases} i\psi_t + 0.5\psi_{xx} + |\psi|^2\psi = 0 & (x,t) \in \Omega \\ \psi(0,x) = 2\,\text{sech}(x) & x \in [-5,5] \\ \psi(t,-5) = \psi(t,5) & t \in (0,T] \\ \psi_x(t,-5) = \psi_x(t,5) & t \in (0,T] \end{cases} \tag{11}$$

其中$T = \pi/2$。

为了评估PINN的精度，Raissi等人 [143] 通过使用传统光谱方法模拟薛定谔方程创建了一个高分辨率数据集。作者使用基于MATLAB的Chebfun开源软件 [40]将薛定谔方程积分到最终时间 $T = \pi/2$ 。

物理信息神经网络（PINN）的解是在测量数据的一个子集上训练的，该子集包括初始数据、边界数据和域内的配置点。初始时间数据，$t = 0$，是

$\{x_0^i, \psi_0^i\}_{i=1}^{N_0}$, the boundary collocation points are $\{t_b^i\}_{i=1}^{N_b}$, and the collocation points on $\mathcal{F}(t, x)$ are $\{t_c^i, x_c^i\}_{i=1}^{N_c}$. In Raissi et al [143] a total of $N_0 = 50$ initial data points on $\psi(x, 0)$ are randomly sampled from the whole high-resolution data-set to be included in the training set, as well as $N_b = 50$ randomly sampled boundary points to enforce the periodic boundaries. Finally for the solution domain, it is assumed $N_c = 20\ 000$ randomly sampled collocation points.

The neural network architecture has two inputs, one for the time t and the other one the location x, while the output has also length 2 rather than 1, as it would normally be assumed, because the output of this NN is expected to find the real and imaginary parts of the solution.

The network is trained in order to minimize the losses due to the initial and boundary conditions, $\mathcal{L}_\mathcal{B}$, as well as to satisfy the Schrodinger equation on the collocation points, i.e. $\mathcal{L}_\mathcal{F}$. Because we are interested in a model that is a surrogate for the PDE, no extra data, $\mathcal{L}_{data} = 0$, is used. In fact we only train the PINN with the known data point from the initial time step $t = 0$. So the losses of (6) are:

$$\begin{aligned}\mathcal{L}_\mathcal{B} &= \frac{1}{N_0}\sum_{i=1}^{N_0}|\psi(0, x_0^i) - \psi_0^i|^2 \\ &+ \frac{1}{N_b}\sum_{i=1}^{N_b}\left(|\psi^i(t_b^i, -5) - \psi^i(t_b^i, 5)|^2 + |\psi_x^i(t_b^i, -5) - \psi_x^i(t_b^i, 5)|^2\right)\end{aligned}$$

and

$$\mathcal{L}_\mathcal{F} = \frac{1}{N_c}\sum_{i=1}^{N_c}|\mathcal{F}(t_c^i, x_c^i)|^2.$$

The Latin Hypercube Sampling technique [164] is used to create all randomly sampled points among the benchmark data prior to training the NN.

In our training we use Adam, with a learning rate of 10^{-3}, followed by a final fine-tuning with LBFGS. We then explore different settings and architectures as in Table 2, by analysing the Mean Absolute Error (MAE) and Mean Squared Error (MSE). We used the PyTorch implementation from Stiller et al [167] which is accessible on GitHub. While the benchmark solutions are from the GitHub of Raissi et al [143].

Raissi et al [143] first used a DNN with 5 layers each with 100 neurons per layer and a hyperbolic tangent activation function in order to represent the unknown function ψ for both real and imaginary parts. In our test, we report the original configuration but we also analyze other network architectures and training point amounts.

A comparison of the predicted and exact solutions at three different temporal snapshots is shown in Figs. 3, and 4. All the different configurations reported in Table 2 show similar patterns of Fig. 4, the only difference is in the order of magnitude of the error.

In particular in such Figure, we show the best configuration obtained in our test, with an average value of $5, 17 \cdot 10^{-04}$ according to the MSE. Figure 4 first shows the modulus of the predicted spatio-temporal solution $|\psi(x, t)|$, with respect to the benchmark solution, by plotting the error at each point. This PINN has some difficulty predicting the central height around $(x, t) = (0, \pi/4)$, as well as in mapping value on in $t \in (\pi/4, \pi/2)$ that are symmetric to the time interval $t \in (0, \pi/4)$.

Finally, in Table 2, we present the results of a few runs in which we analyze what happens to the training loss, relative L_2, MAE, and MSE when we vary the boundary data, and initial

{位置$_0^i$, $\psi_0^i\}^{N_0}{}_{i=1}$, 边界配置点是 {时间$_b^i\}^{N_b}{}_{i=1}$, 而在 $\mathcal{F}$(时间, 位置) 上的配置点是 {时间$_c^i$, 位置$_c^i\}^{N_c}{}_{i=1}$. 在Raissi等人 [143] 中，总共从整个高分辨率数据集中随机采样 $N_0 = 50$ 个初始数据点用于训练集，以及 $N_b = 50$ 个随机采样的边界点以强制执行周期边界。最后，对于解域，假设 $N_c = 20\ 000$ 个随机采样的配置点。

神经网络架构有两个输入，一个用于时间 时间，另一个用于位置 位置，而输出也有长度2而不是1，因为通常情况下会假设输出长度为1，因为该NN的输出预期是找到解的实部和虚部。

网络被训练以最小化由于初始和边界条件引起的损失，$\mathcal{L}_\mathcal{B}$，以及满足配置点上的薛定谔方程，即$\mathcal{L}_\mathcal{F}$。因为我们感兴趣的是一个PDE的替代模型，所以没有使用额外数据，$\mathcal{L}_{数据} = 0$，实际上我们仅用初始时间步的已知数据点训练PINN$t\ = 0$。所以（6）的损失是：

$$\begin{aligned}\mathcal{L}_\mathcal{B} &= \frac{1}{N_0}\sum_{i=1}^{N_0}|\psi(0, x_0^i) - \psi_0^i|^2 \\ &+ \frac{1}{N_b}\sum_{i=1}^{N_b}\left(|\psi^i(t_b^i, -5) - \psi^i(t_b^i, 5)|^2 + |\psi_x^i(t_b^i, -5) - \psi_x^i(t_b^i, 5)|^2\right)\end{aligned}$$

and

$$\mathcal{L}_\mathcal{F} = \frac{1}{N_c}\sum_{i=1}^{N_c}|\mathcal{F}(t_c^i, x_c^i)|^2.$$

拉丁超立方抽样技术 [164] 用于在训练神经网络之前在基准数据中创建所有随机采样的点。

在我们的训练中，我们使用Adam，学习率为 10^{-3}，然后使用LBFGS进行最终微调。我们探索了表2中不同的设置和架构，通过分析平均绝对误差（MAE）和均方误差（MSE）。我们使用了Stiller等人 [167]的PyTorch实现，该实现可在GitHub上获取。基准解来自Raissi等人 [143]的GitHub。

拉西等人 [143] 首次使用一个包含5层、每层100个神经元的深度神经网络，并采用双曲正切激活函数来表示未知函数 ψ 的实部和虚部。在我们的测试中，我们报告了原始配置，但也分析了其他网络架构和训练点数量。

预测解与精确解在三个不同时间快照下的比较如图 3、4所示。表 2 中报告的所有不同配置都显示出与图 4相似的规律，唯一的不同在于误差的数量级。

特别是在该图中，我们展示了测试中获得的最佳配置，其均方误差的平均值为5, $17 \cdot 10^{-04}$ 。图 4 首先展示了预测时空解的模量 $|\psi(x,t)|$相对于基准解的误差，通过在每个点绘制误差。这个物理信息神经网络在预测 $(x,t) = (0, \pi/4)$ 附近的中心高度时存在一些困难，以及在映射对称于时间区间 $t\ \in (0, \pi/4)$的 $t\ \in (\pi/4, \pi/2)$ 上的值时也是如此。

最后，在表<code style id='1'>2</code>中，我们展示了几个运行结果，其中分析了当改变边界数据<code>边界数据</code>和<code>初始条件</code>时，训练损失、相对<code style id='3'>L</code><sup>2</sup>、MAE和MSE的变化情况

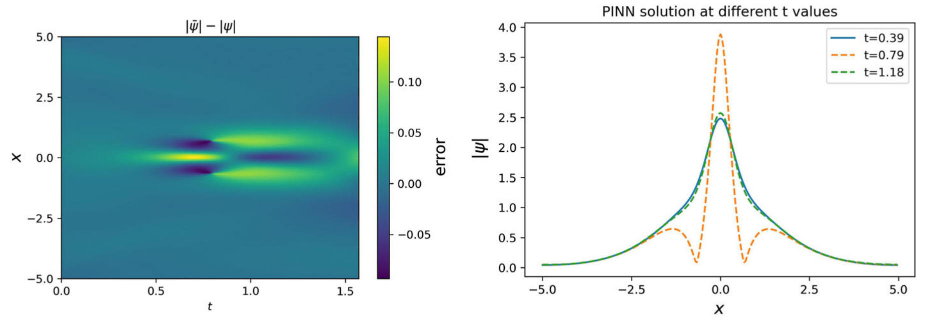

Fig. 4 PINN solutions of the 1D nonlinear Shrödinger (NLS) problem. As illustrated in the first image, the error is represented as the difference among the benchmark solution and PINN result, whereas the second image depicts the PINN's solution at three independent time steps: $t = \pi/8, \pi/4, 3/8\pi$. In this case, we would have expected the solution to overlap at both $\pi/8$ and $3/8\pi$, but there isn't a perfect adherence; in fact, the MSE is $5, 17 \cdot 10^{-04}$ in this case, with some local peak error of 10^{-01} in the second half of the domain

value data. In addition, we can examine how the error grows as the solution progresses through time.

4 PINNs: Data, Applications and Software

The preceding sections cover the neural network component of a PINN framework and which equations have been addressed in the literature. This section first discusses the physical information aspect, which is how the data and model are managed to be effective in a PINN framework. Following that, we'll look at some of the real-world applications of PINNs and how various software packages such as DeepXDE, NeuralPDE, NeuroDiffEq, and others were launched in 2019 to assist with PINNs' design.

4.1 Data

The PINN technique is based not only on the mathematical description of the problems, embedded in the loss or the NN, but also on the information used to train the model, which takes the form of training points, and impacts the quality of the predictions. Working with PINNs requires an awareness of the challenges to be addressed, i.e. knowledge of the key constitutive equations, and also experience in Neural Network building.

Moreover, the learning process of a PINN can be influenced by the relative magnitudes of the different physical parameters. For example to address this issue, Kissas et al [81] used a non-dimensionalization and normalization technique.

However, taking into account the geometry of the problem can be done without effort. PINNs do not require fixed meshes or grids, providing greater flexibility in solving high-dimensional problems on complex-geometry domains.

The training points for PINNs can be arbitrarily distributed in the spatio-temporal domain [128]. However, the distribution of training points influences the flexibility of PINNs. Increasing the number of training points obviously improves the approximation, although in some applications, the location of training places is crucial. Among the various methods for select-

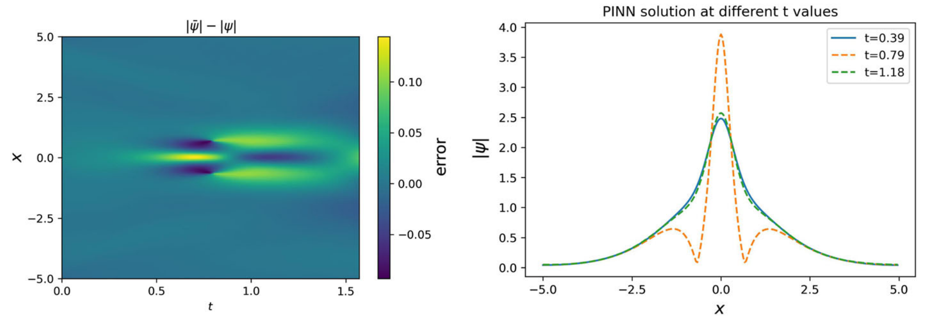

Fig. 4 PINN solutions of the 1D nonlinear Shrödinger (NLS) problem. As illustrated in the first image, the error is represented as the difference among the benchmark solution and PINN result, whereas the second image depicts the PINN's solution at three independent time steps: $t = \pi/8, \pi/4, 3/8\pi$. In this case, we would have expected the solution to overlap at both $\pi/8$ and $3/8\pi$, but there isn't a perfect adherence; in fact, the MSE is $5, 17 \cdot 10^{-04}$ in this case, with some local peak error of 10^{-01} in the second half of the domain

数值数据。此外，我们还可以检查误差如何随解随时间推移而增长。

4 物理信息神经网络：数据、应用和软件

前面的章节涵盖了物理信息神经网络框架中的神经网络组件以及文献中已解决的方程。本节首先讨论物理信息方面，即数据和管理模型如何在物理信息神经网络框架中有效。随后，我们将查看物理信息神经网络的现实世界应用，以及 DeepXDE、NeuralPDE、NeuroDiffEq 等各种软件包如何在 2019 年发布以协助物理信息神经网络的设计。

4.1 数据

PINN技术不仅基于问题的数学描述，嵌入在损失或神经网络中，还基于用于训练模型的信息，即训练点，并影响预测质量。使用PINNs需要了解要解决的问题的挑战，即了解关键本构方程，以及神经网络构建的经验。

此外，PINN的学习过程可能受到不同物理参数相对大小的影响。例如，为了解决这个问题，基萨斯等人 [81] 使用了无量纲化和归一化技术。

然而，考虑问题的几何形状可以毫不费力地完成。PINNs不需要固定的网格或网格，在复杂几何域上解决高维问题提供了更大的灵活性。

PINNs的训练点可以在时空域中任意分布[128]。然而，训练点的分布会影响PINNs的灵活性。增加训练点的数量显然可以提高近似度，尽管在一些应用中，训练位置的位置至关重要。在选择各种方法中，

Table 2 Two subtables exploring the effect of the amount of data points on convergence for PINN and the inherent problems of vanilla pINN to adhere to the solution for longer time intervals, in solving the 1D nonlinear Shrödinger (NLS) Eq. (11)

N_0	N_b	N_c	Training loss	relative L_2	MAE	MSE
(a) Case where the NN is fixed with 100 neurons per layer and four hidden layers. Only the number the number of training points on boundary conditions are doubled						
40	50	20000	9×10^{-4}	0.025 ± 0.003	0.065 ± 0.004	0.019 ± 0.002
40	100	20000	1×10^{-3}	0.024 ± 0.002	0.065 ± 0.003	0.019 ± 0.002
80	50	20000	6×10^{-4}	0.007 ± 0.001	0.035 ± 0.004	0.005 ± 0.001
80	100	20000	6×10^{-4}	0.006 ± 0.002	0.033 ± 0.005	0.005 ± 0.002

time t	relative L_2	MAE	MSE
(b) Error behavior at various time steps for the best occurrence in Table, when Table 2a, where $N_0 = 80$, $N_b = 100$, $N_c = 20000$			
0	$(5 \pm 1) \times 10^{-4}$	0.012 ± 0.002	$(4 \pm 1) \times 10^{-4}$
0.39	$(15{\pm}5) \times 10^{-4}$	0.015 ± 0.002	$(12 \pm 4) \times 10^{-3}$
0.79	0.009 ± 0.003	0.038 ± 0.006	0.007 ± 0.003
1.18	0.01 ± 0.004	0.051 ± 0.009	0.009 ± 0.003
1.56	0.005 ± 0.001	0.044 ± 0.005	0.004 ± 0.001

In Table 2a, the NN consists of four layers, each of which contains 100 neurons, and we can observe how increasing the number of training points over initial or boundary conditions will decrease error rates. Furthermore, doubling the initial condition points has a much more meaningful influence impact than doubling the points on the spatial daily domain in this problem setup. In Table 2b, diven the best NN, we can observe that a vanilla PINN has difficulties in maintaining a strong performance overall the whole space-time domain, especially for longer times, this issue is a matter of research discussed in 5.3. All errors, MAE, MSE and L2-norm Rel are average over 10 runs. For all setups, the same optimization parameters are used, including training with 9000 epochs using Adam with a learning rate of 0.001 and a final L-BFGS fine-tuning step

Table 2 Two subtables exploring the effect of the amount of data points on convergence for PINN and the inherent problems of vanilla pINN to adhere to the solution for longer time intervals, in solving the 1D nonlinear Shrödinger (NLS) Eq. (11)

N_0	N_b	N_c	Training loss	relative L_2	MAE	MSE
(a) Case where the NN is fixed with 100 neurons per layer and four hidden layers. Only the number the number of training points on boundary conditions are doubled						
40	50	20000	9×10^{-4}	0.025 ± 0.003	0.065 ± 0.004	0.019 ± 0.002
40	100	20000	1×10^{-3}	0.024 ± 0.002	0.065 ± 0.003	0.019 ± 0.002
80	50	20000	6×10^{-4}	0.007 ± 0.001	0.035 ± 0.004	0.005 ± 0.001
80	100	20000	6×10^{-4}	0.006 ± 0.002	0.033 ± 0.005	0.005 ± 0.002

time t	relative L_2	MAE	MSE
(b) Error behavior at various time steps for the best occurrence in Table, when Table 2a, where $N_0 = 80$, $N_b = 100$, $N_c = 20000$			
0	$(5 \pm 1) \times 10^{-4}$	0.012 ± 0.002	$(4 \pm 1) \times 10^{-4}$
0.39	$(15{\pm}5) \times 10^{-4}$	0.015 ± 0.002	$(12 \pm 4) \times 10^{-3}$
0.79	0.009 ± 0.003	0.038 ± 0.006	0.007 ± 0.003
1.18	0.01 ± 0.004	0.051 ± 0.009	0.009 ± 0.003
1.56	0.005 ± 0.001	0.044 ± 0.005	0.004 ± 0.001

In Table 2a, the NN consists of four layers, each of which contains 100 neurons, and we can observe how increasing the number of training points over initial or boundary conditions will decrease error rates. Furthermore, doubling the initial condition points has a much more meaningful influence impact than doubling the points on the spatial daily domain in this problem setup. In Table 2b, diven the best NN, we can observe that a vanilla PINN has difficulties in maintaining a strong performance overall the whole space-time domain, especially for longer times, this issue is a matter of research discussed in 5.3. All errors, MAE, MSE and L2-norm Rel are average over 10 runs. For all setups, the same optimization parameters are used, including training with 9000 epochs using Adam with a learning rate of 0.001 and a final L-BFGS fine-tuning step

ing training points, Pang et al [128] addressed lattice-like sampling, i.e. equispaced, and quasi-random sequences, such as the Sobol sequences or the Latin hypercube sampling.

Another key property of PINN is its ability to describe latent nonlinear state variables that are not observable. For instance, Zhang et al [197] observed that when a variable's measurement was missing, the PINN implementation was capable of accurately predicting that variable.

4.2 Applications

In this section, we will explore the real-world applications of PINNs, focusing on the positive leapfrog applications and innovation that PINNs may bring to our daily lives, as well as the implementation of such applications, such as the ability to collect data in easily accessible locations and simulate dynamics in other parts of the system, or applications to hemodynamics flows, elastic models, or geoscience.

Hemodynamics

Three flow examples in hemodynamics applications are presented in Sun et al [168], for addressing either stenotic flow and aneurysmal flow, with standardized vessel geometries and varying viscosity. In the paper, they not only validate the results against CFD benchmarks, but they also estimate the solutions of the parametric Navier–Stokes equation without labeled data by designing a DNN that enforces the initial and boundary conditions and training the DNN by only minimizing the loss on conservation equations of mass and momentum.

In order to extract velocity and pressure fields directly from images, Raissi et al [147] proposed the Hidden fluid mechanics (HFM) framework. The general structure could be applied for electric or magnetic fields in engineering and biology. They specifically apply it to hemodynamics in a three-dimensional intracranial aneurysm.

Patient-specific systemic artery network topologies can make precise predictions about flow patterns, wall shear stresses, and pulse wave propagation. Such typologies of systemic artery networks are used to estimate Windkessel model parameters [81]. PINN methodology is applied on a simplified form of the Navier–Stokes equations, where a hyperbolic conservation law defines the evolution of blood velocity and cross-sectional area instead of mass and momentum conservation. Kissas et al [81] devise a new method for creating 3D simulations of blood flow: they estimate pressure and retrieve flow information using data from medical imaging. Pre-trained neural network models can quickly be changed to a new patient condition. This allows Windkessel parameters to be calculated as a simple post-processing step, resulting in a straightforward approach for calibrating more complex models. By processing noisy measurements of blood velocity and wall displacement, Kissas et al [81] present a physically valid prediction of flow and pressure wave propagation derived directly from non-invasive MRI flow. They train neural networks to provide output that is consistent with clinical data.

Flows Problems

Mathews et al [106] observe the possibility to infer 3D turbulent fields from only 2D data. To infer unobserved field dynamics from partial observations of synthetic plasma, they simulate the drift-reduced Braginskii model using physics-informed neural networks (PINNs) trained to solve supervised learning tasks while preserving nonlinear partial differential equations.

通过设置训练点，Pang等人 [128] 研究了格状采样，即等距采样，以及准随机序列，如索博尔序列或拉丁超立方抽样。

PINN的另一个关键特性是其描述不可观测的潜在非线性状态变量的能力。例如，张等人 [197] 观察到，当变量的测量值缺失时，PINN实现能够准确预测该变量。

4.2 应用

在本节中，我们将探讨PINNs在现实世界中的应用，重点关注正向蛙跳应用以及PINNs可能为我们的日常生活带来的创新，以及此类应用的实现，例如收集易于访问位置的数据和在系统的其他部分模拟动力学，或应用于血流动力学流动、弹性模型或地球科学。

血流动力学

孙等人 [168]，在血流动力学应用中提出了三种流动示例，用于解决狭窄流和动脉瘤流问题，具有标准化的血管几何形状和变化的粘度。在论文中，他们不仅将结果与计算流体动力学基准进行验证，而且还通过设计一个强制执行初始和边界条件的深度神经网络，在没有标记数据的情况下估计参数化纳维-斯托克斯方程的解，并通过仅最小化质量和动量守恒方程的损失来训练深度神经网络。

为了直接从图像中提取速度和压力场，Raissi等人 [147] 提出了隐藏流体力学（HFM）框架。该通用结构可以应用于工程和生物学中的电场或磁场。他们将该方法特别应用于三维颅内动脉瘤的血流动力学。

患者特定的全身动脉网络拓扑可以精确预测流动模式、壁面剪切应力和脉搏波传播。这种类型的全身动脉网络用于估计Windkessel模型参数 [81]。物理信息神经网络方法应用于纳维-斯托克斯方程的简化形式，其中双曲守恒定律定义了血液速度和横截面积的演变，而不是质量和动量守恒。基萨斯等人 [81] 设计了一种新的方法来创建血流3D模拟：他们使用来自医学成像的数据估计压力并检索流动信息。预训练的神经网络模型可以快速更改为新患者条件。这使得Windkessel参数可以作为一个简单的后处理步骤来计算，从而为校准更复杂的模型提供了一种直接的方法。通过处理血液速度和壁位移的噪声测量，基萨斯等人 [81] 提出了一个从无创MRI血流直接推导出的物理有效的流动和压力波传播预测。他们训练神经网络以提供与临床数据一致输出。

流动问题

马修斯等人 [106] 观察到仅从2D数据中推断3D湍流场的可能性。为了从合成等离子体的部分观测中推断未观测场的动力学，他们使用物理信息神经网络（PINNs）模拟了漂移减少的布拉金斯基模型，该模型被训练以解决监督学习任务，同时保留非线性偏微分方程。

This paradigm is applicable to the study of quasineutral plasmas in magnetized collisional situations and provides paths for the construction of plasma diagnostics using artificial intelligence. This methodology has the potential to improve the direct testing of reduced turbulence models in both experiment and simulation in ways previously unattainable with standard analytic methods. As a result, this deep learning technique for diagnosing turbulent fields is simply transferrable, allowing for systematic application across magnetic confinement fusion experiments. The methodology proposed in Mathews et al [106] can be adapted to many contexts in the interdisciplinary research (both computational and experimental) of magnetized collisional plasmas in propulsion engines and astrophysical environments.

Xiao et al [186] examine existing turbulent flow databases and proposes benchmark datasets by methodically changing flow conditions.

In the context of high-speed aerodynamic flows, Mao et al [103], investigates Euler equations solutions approximated by PINN. The authors study both forward and inverse, 1D and 2D, problems. As for inverse problems, they analyze two sorts of problems that cannot be addressed using conventional approaches. In the first problem they determine the density, velocity, and pressure using data from the density gradient; in the second problem, they determine the value of the parameter in a two-dimensional oblique wave equation of state by providing density, velocity, and pressure data.

Projecting solutions in time beyond the temporal area used in training is hard to address with the vanilla version of PINN, and such problem is discussed and tested in Kim et al [79]. The authors show that vanilla PINN performs poorly on extrapolation tasks in a variety of Burges' equations benchmark problems, and provide a novel NN with a different training approach.

PINN methodology is also used also to address the 1D Buckley–Leverett two-phase problem used in petroleum engineering that has a non-convex flow function with one inflection point, making the problem quite complex [1]. Results are compared with those obtained by the Lagrangian–Eulerian and Lax–Friedrichs schemes.

Finally, Buckley–Leverett's problem is also addressed in Almajid and Abu-Al-Saud [4], where PINN is compared to an ANN without physical loss: when only early-time saturation profiles are provided as data, ANN cannot predict the solution.

Optics and Electromagnetic Applications

The hybrid PINN from Fang [44], based on CNN and local fitting method, addresses applications such as the 3-D Helmholtz equation, quasi-linear PDE operators, and inverse problems. Additionally, the author tested his hybrid PINN on an icosahedron-based mesh produced by Meshzoo for the purpose of solving surface PDEs.

A PINN was also developed to address power system applications [114] by solving the swing equation, which has been simplified to an ODE. The authors went on to expand on their research results in a subsequent study [165].

In [85] a whole class of microelectromagnetic problems is addressed with a single PINN model, which learns the solutions of classes of eigenvalue problems, related to the nucleation field associated with defects in magnetic materials.

In Chen et al [30], the authors solve inverse scattering problems in photonic metamaterials and nano-optics. They use PINNs to retrieve the effective permittivity characteristics of a number of finite-size scattering systems involving multiple interacting nanostructures and multi-component nanoparticles. Approaches for building electromagnetic metamaterials are explored in Fang and Zhan [45]. E.g. cloaking problem is addressed in both Fang and

该范式适用于磁化碰撞情况中准中性等离子体的研究，并为使用人工智能构建等离子体诊断方法提供了途径。这种方法有可能以标准分析方法先前无法实现的方式改进实验和模拟中湍流模型的直接测试。因此，这种用于诊断湍流场的深度学习技术可以简单地迁移，允许在磁约束聚变实验中进行系统应用。Mathews 等人 [106] 中提出的这种方法可以适应磁化碰撞等离子体在推进发动机和天体物理环境中的跨学科研究（计算和实验）的许多背景。

萧等人 [186] 检查现有的湍流流动数据库，并通过系统地改变流动条件提出基准数据集。

在高速空气动力学流动的背景下，毛等人 [103]，研究了由物理信息神经网络近似的欧拉方程解。作者研究了正向和逆向、1D 和 2D 的问题。对于逆问题，他们分析了两种无法使用传统方法解决的问题。在第一个问题中，他们使用密度梯度数据来确定密度、速度和压力；在第二个问题中，他们通过提供密度、速度和压力数据来确定二维斜波状态方程中的参数值。

将解在时间上投影到训练中使用的时域之外，使用PINN原版很难解决，这个问题在金等人 [79]中被讨论和测试。作者们表明，PINN原版在各种Burges方程基准问题上在extrapolation任务上的表现很差，并提供了一种具有不同训练方法的新型神经网络。

PINN方法也被用于解决石油工程中使用的1D Buckley–Leverett两相问题，该问题具有非凸的流函数和一个拐点，使得问题相当复杂 [1]。结果与使用拉格朗日-欧拉和拉克斯-弗里德里希斯格式得到的结果进行了比较。

最后，Buckley–Leverett问题在Almajid和Abu-Al-Saud [4],中被解决，其中PINN与没有物理损失的ANN进行了比较：当仅提供早期时间的饱和度分布作为数据时，ANN无法预测解。

光学和电磁应用

方 [44], 基于卷积神经网络和局部拟合方法的混合PINN，解决了三维亥姆霍兹方程、拟线性偏微分算子和逆问题等应用。此外，作者使用Meshzoo生成的基于二十面体的网格测试了他的混合PINN，目的是求解表面偏微分方程。

还开发了一种PINN来解决电力系统应用 [114] ，通过求解已被简化为常微分方程的摆动方程。作者在后续研究中进一步扩展了他们的研究成果 [165]。

在 [85] 中，单个PINN模型解决了一整类微电磁问题，该模型学习了一类特征值问题的解，这些特征值问题与磁性材料缺陷相关的成核场有关。

在陈等人 [30], 中，作者在光子超材料和纳米光学中解决了逆散射问题。他们使用PINNs检索了涉及多个相互作用纳米结构和多组分纳米粒子的有限尺寸散射系统的有效介电特性。在方和战 [45]中探索了构建电磁超材料的方法。例如，隐形问题在方和战中都得到了解决。

Zhan [45], Chen et al [30]. A survey of DL methodologies, including PINNs, applied to nanophotonics may be found in Wiecha et al [183].

As a benchmark problem for DeepM&Mnet, Cai et al [21] choose electroconvection, which depicts electrolyte flow driven by an electric voltage and involves multiphysics, including mass, momentum, cation/anion transport, and electrostatic fields.

Molecular Dynamics and Materials Related Applications

Long-range molecular dynamics simulation is addressed with multi-fidelity PINN (MPINN) by estimating nanofluid viscosity over a wide range of sample space using a very small number of molecular dynamics simulations Islam et al [69]. The authors were able to estimate system energy per atom, system pressure, and diffusion coefficients, in particular with the viscosity of argon-copper nanofluid. PINNs can recreate fragile and non-reproducible particles, as demonstrated by Stielow and Scheel [166], whose network reconstructs the shape and orientation of silver nano-clusters from single-shot scattering images. An encoder–decoder architecture is used to turn 2D images into 3D object space. Following that, the loss score is computed in the scatter space rather than the object space. The scatter loss is calculated by computing the mean-squared difference error between the network prediction and the target's dispersion pattern. Finally, a binary loss is applied to the prediction in order to reinforce the physical concept of the problem's binary nature. They also discover novel geometric shapes that accurately mimic the experimental scattering data.

A multiscale application is done in Lin et al [95, 96], where the authors describe tiny bubbles at both the continuum and atomic levels, the former level using the Rayleigh–Plesset equation and the latter using the dissipative particle dynamics technique. In this paper, the DeepONet architecture [99] is demonstrated to be capable of forecasting bubble growth on the fly across spatial and temporal scales that differ by four orders of magnitude.

Geoscience and Elastostatic Problems

Based on Föppl–von Kármán (FvK) equations, Li et al [94] test their model to four loading cases: in-plane tension with non-uniformly distributed stretching forces, central-hole in-plane tension, deflection out-of-plane, and compression buckling. Moreover stress distribution and displacement field in solid mechanics problems was addressed by Haghighat and Juanes [55].

For earthquake hypocenter inversion, Smith et al [163] use Stein variational inference with a PINN trained to solve the Eikonal equation as a forward model, and then test the method against a database of Southern California earthquakes.

Industrial Application

A broad range of applications, particularly in industrial processes, extends the concept of PINN by adapting to circumstances when the whole underlying physical model of the process is unknown.

The process of lubricant deterioration is still unclear, and models in this field have significant inaccuracies; this is why Yucesan and Viana [194] introduced a hybrid PINN for main bearing fatigue damage accumulation calibrated solely through visual inspections. They use a combination of PINN and ordinal classifier called discrete ordinal classification (DOrC) approach. The authors looked at a case study in which 120 wind turbines were inspected once

Zhan [45], 陈等人 [30]。关于深度学习方法（包括物理信息神经网络）在纳米光子学中的应用的综述可参见Wiecha等人 [183]。

作为DeepM&Mnet的基准问题，Caietal [21] 选择了电对流，它描绘了由电电压驱动的电解质流动，并涉及多物理场，包括质量、动量、阳离子/阴离子传输和静电场。

分子动力学和材料相关应用

长程分子动力学模拟通过多保真物理信息神经网络（MPINN）解决，通过使用极少数分子动力学模拟，在样本空间范围内估计纳米流体的粘度，Islam等人 [69]。作者能够估计每个原子的系统能量、系统压力和扩散系数，特别是氩铜纳米流体的粘度。物理信息神经网络可以重现脆弱且不可重复的粒子，如Stielow和Scheel [166], 所示，其网络从单次散射图像重建银纳米团簇的形状和方向。使用编码器-解码器架构将2D图像转换为3D对象空间。随后，在散射空间而不是对象空间中计算损失分数。通过计算网络预测与目标分散模式之间的均方差误差来计算散射损失。最后，对预测应用二元损失，以加强问题的二元物理概念。他们还发现了能够准确模拟实验散射数据的几何形状。

林等人 [95, 96]，在其中进行了一项多尺度应用，作者们描述了在连续体和原子尺度上的微小气泡，前者使用瑞利-普莱斯特方程，后者使用耗散粒子动力学技术。在本文中，深度在线网络架构 [99] 被证明能够跨时空尺度（差异达四个数量级）实时预测气泡生长。

地球科学和弹性静力学问题

基于弗普尔-卡尔曼方程，李等人 [94] 测试了他们的模型在四种加载情况下的表现：平面内拉伸（非均匀分布的拉伸力）、中心孔平面内拉伸、平面外弯曲和压缩屈曲。此外，哈吉加特和胡安内斯 [55] 解决了固体力学问题中的应力分布和位移场。

对于地震震源反演，史密斯等人 [163] 使用斯坦变分推理，并使用一个训练有素的物理信息神经网络作为正向模型来求解艾克顿方程，然后测试该方法对南加州地震数据库。

工业应用

在工业过程中，应用范围广泛，通过适应整个过程的基础物理模型未知的情况，扩展了物理信息神经网络（PINN）的概念。

润滑剂退化的过程仍然不清楚，该领域的模型存在显著的不准确性；这就是为什么Yucesan和Viana [194] 引入了一种混合PINN用于主轴承疲劳损伤累积，该PINN仅通过视觉检查进行校准。他们使用PINN和序数分类器（称为离散序数分类（DOrC）方法）的组合。作者研究了一个案例研究，其中120台风力涡轮机被检查一次

a month for six months and found the model to be accurate and error-free. The grease damage accumulation model was also trained using noisy visual grease inspection data. Under fairly conservative ranking, researchers can utilize the derived model to optimize regreasing intervals for a particular useful life goal.
As for, corrosion-fatigue crack growth and bearing fatigue are examples studied in Viana et al [174].

For simulating heat transmission inside a channel, Modulus from NVIDIA [61] trains on a parametrized geometry with many design variables. They specifically change the fin dimensions of the heat sink (thickness, length, and height) to create a design space for various heat sinks [123]. When compared to traditional solvers, which are limited to single geometry simulations, the PINN framework can accelerate design optimization by parameterizing the geometry. Their findings make it possible to perform more efficient design space search tasks for complex systems [22].

4.3 Software

Several software packages, including DeepXDE [100], NVIDIA Modulus (previously SimNet) [61], PyDEns [84], and NeuroDiffEq [28] were released in 2019 to make training PINNs easier and faster. The libraries all used feed-forward NNs and the automatic differentiation mechanism to compute analytical derivatives necessary to determine the loss function. The way packages deal with boundary conditions, whether as a hard constraint or soft constraint, makes a significant difference. When boundary conditions are not embedded in the NN but are included in the loss, various losses must be appropriately evaluated. Multiple loss evaluations and their weighted sum complicate hyper-parameter tuning, justifying the need for such libraries to aid in the design of PINNs.

More libraries have been built in recent years, and others are being updated on a continuous basis, making this a dynamic field of research and development. In this subsection, we will examine each library while also presenting a comprehensive synthesis in Table 3.

DeepXDE

DeepXDE [100] was one of the initial libraries built by one of the vanilla PINN authors. This library emphasizes its problem-solving capability, allowing it to combine diverse boundary conditions and solve problems on domains with complex geometries. They also present residual-based adaptive refinement (RAR), a strategy for optimizing the distribution of residual points during the training stage that is comparable to FEM refinement approaches. RAR works by adding more points in locations where the PDE residual is larger and continuing to add points until the mean residual is less than a threshold limit. DeepXDE also supports complex geometry domains based on the constructive solid geometry (CSG) technique. This package showcases five applications in its first version in 2019, all solved on scattered points: Poisson Equation across an L-shaped Domain, 2D Burgers Equation, first-order Volterra integrodifferential equation, Inverse Problem for the Lorenz System, and Diffusion–Reaction Systems.

Since the package's initial release, a significant number of other articles have been published that make use of it. DeepXDE is used by the authors for: for inverse scattering [30], or deriving mechanical characteristics from materials with MFNNs [98].

While other PINNs are implemented using DeepXDE, like hPINN [78].

六个月的模型训练持续了一个月，并发现模型准确无误。润滑脂损伤累积模型也使用噪声视觉润滑脂检查数据进行了训练。在相当保守的排名下，研究人员可以利用推导出的模型来优化特定使用寿命目标的重新润滑间隔。

至于，腐蚀疲劳裂纹扩展和轴承疲劳是Viana等人 [174] 研究的示例。

为了模拟通道内的热传递，来自英伟达的Modulus [61] 在具有许多设计变量的参数化几何形状上进行训练。他们特别改变了散热器（厚度、长度和高度）的鳍尺寸，以创建各种散热器 [123] 的设计空间。与仅限于单几何形状模拟的传统求解器相比，物理信息神经网络框架可以通过参数化几何形状来加速设计优化。他们的发现使得对复杂系统进行更高效的设计空间搜索任务成为可能 [22]。

4.3 软件

2019年，DeepXDE [100], NVIDIA Modulus（前身为Sim-Net） [61], PyDEns [84], 和NeuroDiffEq [28] 等多个软件包发布，旨在简化并加速物理信息神经网络的训练。这些库均采用前馈神经网络和自动微分机制来计算确定损失函数所需的解析导数。软件包处理边界条件的方式——无论是作为硬约束还是软约束——具有显著差异。当边界条件未嵌入神经网络而包含在损失函数中时，必须适当评估各种损失。多次损失评估及其加权求和使超参数调整变得复杂，这解释了为何需要此类库来辅助物理信息神经网络的构建。

近年来，更多库已被内置，其他库也在持续更新中，这使得该领域成为研究和发展中的动态领域。在本节中，我们将检查每个库，同时在表3中呈现综合概述。

DeepXDE

DeepXDE [100] 是基础物理信息神经网络（vanilla PINN）作者之一构建的初始库之一。该库强调其问题解决能力，允许其组合多种边界条件，并在具有复杂几何形状的域上解决问题。他们还提出了基于残差的自适应细化（RAR），这是一种在训练阶段优化残差点分布的策略，与FEM细化方法相当。RAR通过在PDE残差较大的位置添加更多点，并继续添加点直到平均残差小于阈值限制来工作。DeepXDE还支持基于构造实体几何（CSG）技术的复杂几何形状域。该软件包在其2019年发布的第一个版本中展示了五个应用，所有应用都在散点上进行：L形域上的泊松方程、2D Burgers方程、一阶Volterra积分微分方程、洛伦兹系统的逆问题以及扩散-反应系统。

自该软件包的初始发布以来，已有大量其他文章使用它。作者使用DeepXDE 进行逆散射 [30], 或使用 MFNNs [98] 从材料中推导力学特性。虽然其他物理信息神经网络也是使用 DeepXDE 实现的，例如 hPINN [78]。

Furthermore, more sophisticated tools are built on DeepXDE, such as DeepONets [99], and its later extension DeepM&Mnet [21, 104]. DeepONets and their derivatives are considered by the authors to have the significant potential in approximating operators and addressing multi-physics and multi-scale problems, like inferring bubble dynamics [95, 96].

Finally, DeepXDE is utilized for medical ultrasonography applications to simulate a linear wave equation with a single time-dependent sinusoidal source function [3], and the open-source library is also employed for the Buckley–Leverett problem [4]. A list of research publications that made use of DeepXDE is available online [1]

NeuroDiffEq

NeuroDiffEq [28] is a PyTorch-based library for solving differential equations with neural networks, which is being used at Harvard IACS. NeuroDiffEq solves traditional PDEs (such as the heat equation and the Poisson equation) in 2D by imposing strict constraints, i.e. by fulfilling initial/boundary conditions via NN construction, which makes it a PCNN. They employ a strategy that is similar to the trial function approach [89], but with a different form of the trial function. However, because NeuroDiffEq enforces explicit boundary constraints rather than adding the corresponding losses, they appear to be inadequate for arbitrary bounds that the library does not support [12].

Modulus

Modulus [123], previously known as NVIDIA SimNet [61], from Nvidia, is a toolset for academics and engineers that aims to be both an extendable research platform and a problem solver for real-world and industrial challenges.

It is a PINN toolbox with support to Multiplicative Filter Networks and a gradient aggregation method for larger batch sizes. Modulus also offers Constructive Solid Geometry (CSG) and Tessellated Geometry (TG) capabilities, allowing it to parameterize a wide range of geometries.

In terms of more technical aspects of package implementations, Modulus uses an integral formulation of losses rather than a summation as is typically done. Furthermore, global learning rate annealing is used to fine-tune the weights parameters ω in the loss Eq. 6. Unlike many other packages, Modulus appears to be capable of dealing with a wide range of PDE in either strong or weak form. Additionally, the toolbox supports a wide range of NN, such as Fourier Networks and the DGM architecture, which is similar to the LSTM architecture.

Nvidia showcased the PINN-based code to address multiphysics problems like heat transfer in sophisticated parameterized heat sink geometry [32], 3D blood flow in Intracranial Aneurysm or address data assimilation and inverse problems on a flow passing a 2D cylinder [123]. Moreover, Modulus solves the heat transport problem more quickly than previous solvers.

SciANN

SciANN [55] is an implementation of PINN as a high-level Keras wrapper. Furthermore, the SciANN repository collects a wide range of examples so that others can replicate the results and use those examples to build other solutions, such as elasticity, structural mechanics, and

[1] https://deepxde.readthedocs.io/en/latest/user/research.html.

此外，DeepXDE上构建了更复杂的工具，如DeepONets [99], 及其后续扩展DeepM&Mnet [21, 104]。作者认为DeepONets及其衍生工具在逼近算子和处理多物理和多尺度问题（如推断气泡动力学 [95, 96]）方面具有显著潜力。

最后，DeepXDE被用于医学超声应用，以模拟具有单一时间依赖正弦源函数的线性波方程 [3], 开源库也被用于Buckley–Leverett问题 [4]。使用DeepXDE的研究出版物列表可在网上获取1

NeuroDiffEq

NeuroDiffEq [28] 是一个基于PyTorch的库，用于使用神经网络求解微分方程，它正在哈佛IACS使用。NeuroDiffEq通过施加严格约束（即通过神经网络构建满足初始/边界条件）在2D中求解传统偏微分方程（如热方程和泊松方程），这使得它成为PCNN。他们采用了一种类似于试探函数方法 [89], 的策略，但试探函数的形式不同。然而，因为NeuroDiffEq强制执行显式边界约束而不是添加相应的损失，它们似乎不适用于库不支持的任意边界 [12]。

Modulus

Modulus [123], 之前称为NVIDIA SimNet [61], ，来自Nvidia，是一个面向学术界和工程师的工具集，旨在成为一个可扩展的研究平台，并解决现实世界和工业挑战。

它是一个支持乘法滤波网络和用于较大批量大小的梯度聚合方法的PINN工具箱。Modulus还提供构造实体几何（CSG）和镶嵌几何（TG）功能，允许它参数化各种几何形状。

在软件包实现的技术方面，Modulus使用损失积分公式而不是通常的求和。此外，使用全局学习率退火来微调权重参数 ω 在损失公式6中。与许多其他软件包不同，Modulus似乎能够处理各种偏微分方程，无论是强形式还是弱形式。此外，该工具箱支持各种神经网络，如傅里叶网络和DGM架构，后者类似于LSTM架构。

Nvidia 展示了基于物理信息神经网络（PINN）的代码，用于解决多物理场问题，例如在复杂的参数化散热器几何形状中的热传输、颅内动脉瘤中的 [32], 3D 血流，或解决通过 2D 圆柱的流动的数据同化和逆问题 [123]。此外，Modulus 比之前的求解器更快地解决了热传输问题。

SciANN

SciANN [55] 是物理信息神经网络（PINN）的一个高级 Keras 封装。此外，SciANN 仓库收集了广泛的示例，以便其他人可以复制结果并使用这些示例来构建其他解决方案，例如弹性、结构力学和

1 https://deepxde.readthedocs.io/en/latest/user/research.html.

vibration applications. SciANN is used by the same authors also for creating a nonlocal PINN method in Haghighat et al [56], or for a PINN multi-network model applied on solid mechanics [57]. Although tests for simulating 2D flood, on Shallow Water Equations, are conducted using SciANN [72], the authors wrote the feedforward step into a separate function to avoid the overhead associated with using additional libraries. A general comparison of many types of mesh-free surrogate models based on machine learning (ML) and deep learning (DL) methods is presented in Hoffer et al [64], where SciANN is used among other toolsets. Finally, the PINN framework for solving the Eikonal equation by Waheed et al [175] was implemented using SciAnn.

PyDENs

PyDENs [84] is an open-source neural network PDE solver that allows to define and configure the solution of heat and wave equations. It impose initial/boundary conditions in the NN, making it a PCNN. After the first release in 2019, the development appears to have stopped in 2020.

NeuralPDE.jl

NeuralPDE.jl is part of SciML, a collection of tools for scientific machine learning and differential equation modeling. In particular SciML (Scientific Machine Learning) [137] is a program written in Julia that combines physical laws and scientific models with machine learning techniques.

ADCME

ADCME [187] can be used to develop numerical techniques and connect them to neural networks: in particular, ADCME was developed by extending and enhancing the functionality of TensorFlow. In Xu and Darve [187], ADCME is used to solve different examples, like nonlinear elasticity, Stokes problems, and Burgers' equations. Furthermore, ADCME is used by Xu and Darve [188] for solving inverse problems in stochastic models by using a neural network to approximate the unknown distribution.

Nangs

Nangs [131] is a Python library that uses the PDE's independent variables as NN (inputs), it then computes the derivatives of the dependent variables (outputs), with the derivatives they calculate the PDEs loss function used during the unsupervised training procedure. It has been applied and tested on a 1D and 2D advection-diffusion problem. After a release in 2020, the development appears to have stopped. Although, NeuroDiffEq and Nangs libraries were found to outperform PyDEns in solving higher-dimensional PDEs [134].

TensorDiffEq

TensorDiffEq [107] is a Scientific Machine Learning PINN based toolkit on Tensorflow for Multi–Worker Distributed Computing. Its primary goal is to solve PINNs (inference) and inverse problems (discovery) efficiently through scalability. It implements a Self–Adaptive PINN to increase the weights when the corresponding loss is greater; this task is accomplished by training the network to simultaneously minimize losses and maximize weights.

振动应用。相同的作者还使用SciANN在Haghighat等人 [56], 中创建了一种非局部PINN方法，或在固体力学[57]上应用了PINN多网络模型。尽管使用SciANN [72], 对浅水方程进行的二维洪水模拟测试，但作者将前馈步骤写入了一个单独的函数，以避免使用附加库带来的开销。Hoffer等人 [64], 中介绍了基于机器学习（ML）和深度学习（DL）方法的多种无网格代理模型的通用比较，其中SciANN被用作其他工具集之一。最后，Waheed等人 [175] 使用SciAnn实现的PINN框架用于求解艾克顿方程。

PyDENs

PyDENs [84] 是一个开源的神经网络PDE求解器，允许定义和配置热方程和波方程的解。它在神经网络中施加初始/边界条件，使其成为PCNN。在2019年首次发布后，开发似乎在2020年停止了。

NeuralPDE.jl

NeuralPDE.jl 是 SciML 的一部分，SciML 是一个用于科学机器学习和微分方程建模的工具集合。特别是 SciML（科学机器学习） [137] 是一个用 Julia 编写的程序，它将物理定律和科学模型与机器学习技术相结合。

ADCME

ADCME [187] 可用于开发数值技术并将其与神经网络连接起来：特别是，ADCME 是通过扩展和增强 TensorFlow 的功能而开发的。在 Xu 和 Darve [187], 中，ADCME 用于解决不同的示例，如非线性弹性、Stokes问题和伯格斯方程。此外，Xu 和 Darve [188] 使用神经网络来近似未知分布，通过使用神经网络来解决随机模型中的逆问题。

Nangs

Nangs [131] 是一个 Python 库，它使用 PDE 的独立变量作为 NN（输入），然后计算因变量的导数（输出），他们用这些导数计算在无监督训练程序期间使用的 PDEs 损失函数。它已在 1D 和 2D 对流扩散问题中得到应用和测试。在 2020 年发布后，开发似乎已经停止。尽管，发现 NeuroDiffEq 和 Nangs 库在解决高维 PDEs [134]方面优于 PyDEns。

TensorDiffEq

TensorDiffEq [107] 是一个基于Tensorflow的多工作者分布式计算的科学机器学习PINN工具包。其主要目标是通过可扩展性高效地解决PINNs（推理）和逆问题（发现）。它实现了一种自适应PINN，当相应损失较大时增加权重；这项任务通过训练网络同时最小化损失和最大化权重来完成。

IDRLnet

IDRLnet [132] is a Python's PINN toolbox inspired by Nvidia SimNet [61]. It provides a way to mix geometric objects, data sources, artificial neural networks, loss metrics, and optimizers. It can also solve noisy inverse problems, variational minimization problem, and integral differential equations.

Elvet

Elvet [7] is a Python library for solving differential equations and variational problems. It can solve systems of coupled ODEs or PDEs (like the quantum harmonic oscillator) and variational problems involving minimization of a given functional (like the catenary or geodesics solutions-based problems).

Other Packages

Packages that are specifically created for PINN can not only solve problems using PINN, but they can also be used to provide the groundwork for future research on PINN developments. However, there are other packages that can take advantage of future research developments, such as techniques based on kernel methods [74]. Indeed, rather than switching approaches on optimizers, losses, and so on, an alternative approach with respect to PINN framework is to vary the way the function is represented. Throughout this aspect, rather than using Neural Networks, a kernel method based on Gaussian process could be used. The two most noteworthy Gaussian processes toolkit are the Neural Tangents [122] kernel (NTK), based on JAX, and GPyTorch [49], written using PyTorch. Neural Tangents handles infinite-width neural networks, allowing for the specification of intricate hierarchical neural network topologies. While GPyTorch models Gaussian processes based on Blackbox Matrix–Matrix multiplication using a specific preconditioner to accelerate convergence.

5 PINN Future Challenges and Directions

What comes next in the PINN theoretical or applied setup is unknown. What we can assess here are the paper's incomplete results, which papers assess the most controversial aspects of PINN, where we see unexplored areas, and where we identify intersections with other disciplines.

Although several articles have been published that have enhanced PINNs capabilities, there are still numerous unresolved issues, such as various applications to real-world situations and equations. These span from more theoretical considerations (such as convergence and stability) to implementation issues (boundary conditions management, neural networks design, general PINN architecture design, and optimization aspects). PINNs and other DL methods using physics prior have the potential to be an effective way of solving high-dimensional PDEs, which are significant in physics, engineering, and finance. PINNs, on the other hand, struggle to accurately approximate the solution of PDEs when compared to other numerical methods designed for a specific PDE, in particular, they can fail to learn complex physical phenomena, like solutions that exhibit multi-scale, chaotic, or turbulent behavior.

IDRLnet

IDRLnet [132] 是一个受Nvidia SimNet [61]启发的Python的PINN工具箱。它提供了一种混合几何对象、数据源、人工神经网络、损失指标和优化器的方法。它还可以求解噪声逆问题、变分最小化问题以及积分微分方程。

Elvet

Elvet [7] 是一个用于求解微分方程和变分问题的Python库。它可以求解耦合常微分方程或偏微分方程组（如量子谐振子）以及涉及给定泛函最小化的变分问题（如悬链线或测地线解问题）。

其他软件包

专门为物理信息神经网络（PINN）创建的软件包不仅可以使用PINN解决问题，还可以为未来PINN发展研究提供基础。然而，还有其他软件包可以利用未来的研究发展，例如基于核方法的技术 [74]。实际上，与其在优化器、损失等方面切换方法，关于物理信息神经网络框架的另一种方法是改变函数的表示方式。在这方面，而不是使用神经网络，可以使用基于高斯过程的核方法。最重要的两个高斯过程工具包是JAX基础的神经切线 [122] 核（NTK）和用PyTorch编写的GPyTorch [49], 。神经切线处理无限宽度神经网络，允许指定复杂的分层神经网络拓扑。而GPyTorch使用特定预条件子基于黑盒矩阵-矩阵乘法对高斯过程进行建模，以加速收敛。

5 PINN 未来挑战与方向

PINN 理论或应用设置中的下一步是什么尚不清楚。在这里，我们可以评估论文的不完整结果，这些论文评估了 PINN 最具争议的方面，我们看到了未探索的领域，以及我们确定了与其他学科的交叉点。

尽管已经发表了几篇增强 PINN 能力的文章，但仍有许多未解决的问题，例如将其应用于现实世界情况和方程。这些问题涵盖了更理论性的考虑（如收敛性和稳定性）到实施问题（边界条件管理、神经网络设计、通用 PINN 架构设计和优化方面）。使用物理先验的 PINNs 和其他深度学习方法有潜力成为解决高维偏微分方程的有效方法，这在物理学、工程学和金融学中具有重要意义。然而，与其他为特定偏微分方程设计的数值方法相比，PINNs 在准确近似偏微分方程的解方面存在困难，特别是在它们无法学习复杂的物理现象时，例如表现出多尺度、混沌或湍流行为的解。

Table 3 Major software libraries specifically designed for physics-informed machine learning

Software Name	Backend	Available for	Usage	License	First release	Code Link
DeepXDE Lu et al [100]	TensorFlow, PyTorch, JAX, PaddlePaddle	Python	Solver	Apache-2.0 License	v0.1.0 - Jun 13, 2019	*deepxde*
PyDEns Koryagin et al [84]	Tensorflow	Python	Solver	Apache-2.0 License	v alpha - Jul 14, 2019	*pydens*
NVIDIA Modulus Hennigh et al [61]	PyTorch	Python based API	Solver	Proprietary	v21.06 on Nov 9, 2021	*modulus*
NeuroDiffEq Chen et al [28]	PyTorch	Python	Solver	MIT License	v alpha - Mar 24, 2019	*neurodiffeq*
SciANN Haghighat and Juanes [55]	TensorFlow	Python 2.7–3.6	Wrapper	MIT License	v alpha - Jul 21, 2019	*sciann*
NeuralPDE Zubov et al [201]	Julia	Julia	Solver	MIT License	v0.0.1 - Jun 22, 2019	*NeuralPDE.jl*
ADCME Xu and Darve [187]	Julia TensorFlow	Julia	Wrapper	MIT License	v alpha - Aug 27, 2019	*ADCME.jl*
Nangs Pedro et al [131]	Pytorch	Python	Solver	Apache-2.0 License	v0.0.1 - Jan 9, 2020	*nangs*
TensorDiffEq McClenny et al [107]	Tensorflow 2.x	Python	Solver	MIT License	v0.1.0 - Feb 03, 2021	*TensorDiffEq*
IDRLnet Peng et al [132]	PyTorch, Sympy	Python	Solver	Apache-2.0 License	v0.0.1-alpha Jul 05, 2021	*idrlnet*
Elvet Araz et al [7]	Tensorflow	Python	Solver	MIT License	v0.1.0 Mar 26, 2021	*elvet*

Table 3 Major software libraries specifically designed for physics-informed machine learning

Software Name	Backend	Available for	Usage	License	First release	Code Link
DeepXDE Lu et al [100]	TensorFlow, PyTorch, JAX, PaddlePaddle	Python	Solver	Apache-2.0 License	v0.1.0 - Jun 13, 2019	*deepxde*
PyDEns Koryagin et al [84]	Tensorflow	Python	Solver	Apache-2.0 License	v alpha - Jul 14, 2019	*pydens*
NVIDIA Modulus Hennigh et al [61]	PyTorch	Python based API	Solver	Proprietary	v21.06 on Nov 9, 2021	*modulus*
NeuroDiffEq Chen et al [28]	PyTorch	Python	Solver	MIT License	v alpha - Mar 24, 2019	*neurodiffeq*
SciANN Haghighat and Juanes [55]	TensorFlow	Python 2.7–3.6	Wrapper	MIT License	v alpha - Jul 21, 2019	*sciann*
NeuralPDE Zubov et al [201]	Julia	Julia	Solver	MIT License	v0.0.1 - Jun 22, 2019	*NeuralPDE.jl*
ADCME Xu and Darve [187]	Julia TensorFlow	Julia	Wrapper	MIT License	v alpha - Aug 27, 2019	*ADCME.jl*
Nangs Pedro et al [131]	Pytorch	Python	Solver	Apache-2.0 License	v0.0.1 - Jan 9, 2020	*nangs*
TensorDiffEq McClenny et al [107]	Tensorflow 2.x	Python	Solver	MIT License	v0.1.0 - Feb 03, 2021	*TensorDiffEq*
IDRLnet Peng et al [132]	PyTorch, Sympy	Python	Solver	Apache-2.0 License	v0.0.1-alpha Jul 05, 2021	*idrlnet*
Elvet Araz et al [7]	Tensorflow	Python	Solver	MIT License	v0.1.0 Mar 26, 2021	*elvet*

5.1 Overcoming Theoretical Difficulties in PINN

A PINN can be thought of as a three-part modular structure, with an approximation (the neural network), a module to define how we want to correct the approximation (the physics informed network, i.e. the loss), and the module that manages the minimization of the losses. The NN architecture defines how well the NN can approximate a function, and the error we make in approximating is called approximation error, as seen in Sect. 2.4. Then, how we decide to iteratively improve the approximator will be determined by how we define the loss and how many data points we are integrating or calculating the sum, with the quality of such deviation measured as the generalization error. Finally, the quality of iterations in minimizing the losses is dependent on the optimization process, which gives the optimization error.

All of these factors raise numerous questions for future PINN research, the most important of which is whether or not PINN converges to the correct solution of a differential equation. The approximation errors must tend to zero to achieve stability, which is influenced by the network topology. The outcomes of this research are extremely limited. For example, the relative error for several neural architectures was calculated by altering the number of hidden layers and the number of neurons per layer in Mo et al [115]. In another example, Blechschmidt and Ernst [19] shows the number of successes (i.e. when training loss that is less than a threshold) after ten different runs and for different network topologies (number of layers, neurons, and activation function). Instead, Mishra and Molinaro [111] obtain error estimates and identify possible methods by which PINNs can approximate PDEs. It seems that initial hidden layers may be in charge of encoding low-frequency components (fewer points are required to represent low-frequency signals) and the subsequent hidden layers may be in charge of representing higher-frequency components [105]. This could be an extension of the Frequency-principle, F-principle [198], according to which DNNs fit target functions from low to high frequencies during training, implying a low-frequency bias in DNNs and explaining why DNNs do not generalize well on randomized datasets. For PINN, large-scale features should arise first while small-scale features will require multiple training epochs.

The effects of initialization and loss function on DNN learning, specifically on generalization error, should be investigated. Many theoretical results treat loos estimation based on quadrature rules, on points selected randomly and identically distributed. There are some PINN approaches that propose to select collocations points in specific areas of the space-time domain [118]; this should be investigated as well. Finally, dynamic loss weighting for PINN appears to be a promising research direction [120].

Optimization tasks are required to improve the performances of NNs, which also holds for PINN. However, given the physical integration, this new PINN methodology will require additional theoretical foundations on optimization and numerical analysis, and dynamical systems theory. According to [179, 181], a key issue is to understand the relationship between PDE stiffness and the impact of algorithms as the gradient descent on the PINNs.

Another intriguing research topic is why PINN does not suffer from dimensionality. According to the papers published on PINN, they can scale up easily independently of the size of the problems [112]. The computational cost of PINN does not increase exponentially as the problem dimensionality increases; this property is common to neural network architecture, and there is no formal explanation for such patterns [36]. Bauer and Kohler [13] recently demonstrated that least-squares estimates based on FNN can avoid the curse of dimensionality in nonparametric regression. While Zubov et al [202] demonstrates the capability of the PINN technique with quadrature methods in solving high dimensional problems.

In PINN the process of learning gives rise to a predictor, u_θ, which minimizes the empirical risk (loss). In machine learning theory, the prediction error can be divided into two

5.1 克服物理信息神经网络的理论困难

物理信息神经网络（PINN）可以被视为一个由三部分组成的模块化结构，包括逼近（神经网络）、一个定义我们如何修正逼近的模块（物理信息网络，即损失），以及一个管理损失最小化的模块。NN架构定义了神经网络逼近函数的能力，而我们逼近时产生的误差称为逼近误差，如第2.4节所述。2.4。然后，我们如何决定迭代改进逼近器将取决于我们如何定义损失以及我们集成或计算总和的数据点数量，这种偏差的质量作为泛化误差进行衡量。最后，最小化损失迭代的质量取决于优化过程，这会产生优化误差。

所有这些因素为未来的PINN研究提出了许多问题，其中最重要的是PINN是否收敛到微分方程的正确解。逼近误差必须趋于零以实现稳定性，这受网络拓扑的影响。这项研究的结果非常有限。例如，Mo等人 [115]通过改变隐藏层的数量和每层的神经元数量来计算几种神经架构的相对误差。在另一个例子中，Blechschmidt和Ernst [19] 展示了在十次不同运行和不同网络拓扑（层数、神经元数量和激活函数）下成功的次数（即训练损失小于阈值的次数）。相反，Mishra和Molinaro [111] 获得了误差估计，并确定了PINNs可以逼近偏微分方程（PDEs）的可能方法。据信，初始隐藏层可能负责编码低频分量（表示低频信号所需的点较少），而后续隐藏层可能负责表示高频分量 [105]。这可能是频率原则，F原则 [198], 的扩展，该原则指出深度神经网络（DNNs）在训练过程中从低频到高频拟合目标函数，这意味着DNNs存在低频偏差，并解释了为什么DNNs在随机数据集上泛化能力不佳。对于PINN，大规模特征应首先出现，而小规模特征将需要多个训练周期。

初始化和损失函数对DNN学习的影响，特别是对泛化误差的影响，应该进行调查。许多理论结果基于求积规则，在随机且相同分布的点上进行损失估计。有一些PINN方法提出在时空域的特定区域选择配置点 [118]；这也应该进行调查。最后，PINN的动态损失加权似乎是一个有前景的研究方向 [120]。

优化任务需要提高神经网络的性能，这对PINN也适用。然而，考虑到物理积分，这种新的PINN方法将需要对优化、数值分析和动力系统理论进行额外的理论基础。根据 [179, 181], ，一个关键问题是理解PDE刚度和算法（如梯度下降）对PINNs的影响之间的关系。

另一个有趣的研究课题是为什么PINN不受维度问题的影响。根据关于PINN发表的论文，它们可以轻松扩展，而与问题的规模无关 [112]。PINN的计算成本不会随着问题维度的增加而呈指数增长；这种特性是神经网络架构的常见现象，但没有对这种模式的正式解释 [36]。鲍尔和科尔勒 [13] 最近证明，基于FNN的最小二乘估计可以避免非参数回归中的维度诅咒。而祖博夫等人 [202] 展示了PINN技术在求解高维问题中的能力。

在物理信息神经网络（PINN）中，学习过程会产生一个预测器，u_θ，它最小化经验风险（损失）。在机器学习理论中，预测误差可以分为两部分

components: bias error and variance error. The bias-variance trade-off appears to contradict the empirical evidence of recent machine-learning systems when neural networks trained to interpolate training data produce near-optimal test results. It is known that a model should balance underfitting and overfitting, as in the typical U curve, according to the bias-variance trade-off. However, extremely rich models like neural networks are trained to exactly fit (i.e., interpolate) the data. Belkin et al [14], demonstrate the existence of a double-descent risk curve across a wide range of models and datasets, and they offer a mechanism for its genesis. The behavior of PINN in such a framework of Learning Theory for Deep Learning remains to be investigated and could lead to further research questions. In particular, the function class $\mathcal{H}$ of the hypothesis space in which PINN is optimized might be further examined by specifying such space based on the type of differential equations that it is solving and thus taking into account the physics informed portion of the network.

In general, PINN can fail to approximate a solution, not due to the lack of expressivity in the NN architecture but due to soft PDE constraint optimization problems Krishnapriyan et al [86].

5.2 Improving Implementation Aspects in PINN

When developing a PINN, the PINN designer must be aware that there may be additional configurations that need to be thoroughly investigated in the literature, as well as various tweaks to consider and good practices that can aid in the development of the PINN, by systematically addressing each of the PINN's three modules. From neural network architecture options to activation function type. In terms of loss development, the approaching calculation of the loss integral, as well as the implementation of the physical problem from adimensionalization or the solution space constrains. Finally, the best training procedure should be designed. How to implement these aspects at the most fundamental level, for each physical problem appears to be a matter of ongoing research, giving rise to a diverse range of papers, which we addressed in this survey, and the missing aspects are summarized in this subsection.

Regarding neural networks architectures, there is still a lack of research for non FFNN types, like CNN and RNN, and what involves their theoretical impact on PINNs [181]; moreover, as for the FFNNs many questions remain, like implementations ones regarding the selection of network size [56]. By looking at Table 1, only a small subset of Neural Networks has been instigated as the network architecture for a PINN, and on a quite restricted set of problems. Many alternative architectures are proposed in the DL literature [35], and PINN development can benefit from this wide range of combinations in such research fields. A possible idea could be to apply Fourier neural operator (FNO) [182], in order to learn a generalized functional space [138]. N–BEATS [124], a deep stack of fully connected layers connected with forward and backward residual links, could be used for time series based phenomena. Transformer architecture instead could handle long-range dependencies by modeling global relationships in complex physical problems [75]. Finally, sinusoidal representation networks (SIRENs) Sitzmann et al [161] that are highly suited for expressing complicated natural signals and their derivatives, could be used in PINN. Some preliminary findings are already available [67, 185]. A research line is to study if it is better to increase the breadth or depth of the FFNN to improve PINN outcomes. Some general studies on DNN make different assertions about whether expanding width has the same benefits as increasing depth. This may prompt the question of whether there is a minimum depth/width below which a network cannot understand the physics [173]. The interoperability of PINN will also play an important role in future research [149]. A greater understanding of PINN's activation

组成部分：偏差误差和方差误差。偏差-方差权衡似乎与近期机器学习系统的经验证据相矛盾，当神经网络训练以插值训练数据时会产生接近最优的测试结果。众所周知，根据偏差-方差权衡，模型应平衡欠拟合和过拟合，如典型的U曲线所示。然而，像神经网络这样的极其丰富的模型被训练以精确拟合（即插值）数据。贝尔金等人 [14]，证明了在各种模型和数据集上存在双下降风险曲线，并提出了其产生的机制。在深度学习的学习理论框架下，物理信息神经网络（PINN）的行为仍有待研究，并可能引发进一步的研究问题。特别是，PINN优化的假设空间中的函数类 $\mathcal{H}$ 可能需要通过根据其求解的微分方程类型来指定该空间，从而考虑网络中的物理信息部分进行进一步研究。

通常情况下，PINN无法逼近解，并非由于NN架构的表达能力不足，而是由于软偏微分方程约束优化问题，克里希纳普拉扬等人 [86]。

5.2 改进PINN的实现方面

在开发PINN时，PINN设计者必须意识到，文献中可能存在需要深入研究的额外配置，以及需要考虑的各种调整和有助于PINN开发的好实践，通过系统地解决PINN的三个模块。从神经网络架构选项到激活函数类型。在损失开发方面，损失积分的逼近计算，以及从无量纲化或解空间约束实现物理问题。最后，应设计最佳训练程序。如何在每个物理问题中最基本层面实现这些方面，似乎是一个持续研究的课题，由此产生了各种论文，我们在这篇调查中进行了阐述，缺失的方面在本小节中进行了总结。

关于神经网络架构，对于非前馈神经网络（FFNN）类型，如卷积神经网络（CNN）和循环神经网络（RNN），仍然缺乏研究，以及它们对物理信息神经网络（PINN）的理论影响是什么 [181]；此外，对于前馈神经网络（FFNNs），许多问题仍然存在，例如关于网络规模选择的实现问题 [56]。通过查看表1，只有一小部分神经网络被用作PINN的网络架构，并且在相当有限的问题集上。深度学习文献中提出了许多替代架构 [35]，，PINN的发展可以从这些研究领域的广泛组合中受益。可能的思路是应用傅里叶神经算子（FNO） [182], 来学习广义函数空间 [138]。N–BEATS [124], 是一种由前向和后向残差链接连接的全连接层深度堆栈，可用于基于时间序列的现象。Transformer架构相反，可以通过对复杂物理问题中的全局关系建模来处理长距离依赖 [75]。最后，正弦表示网络（SIRENs）Sitzmann等人 [161] 非常适合表达复杂的自然信号及其导数，可用于PINN。一些初步的发现已经可用 [67, 185]。一个研究方向是研究增加前馈神经网络的宽度或深度是否可以改善PINN结果。关于深度神经网络（DNN）的一些一般性研究表明，扩展宽度与增加深度是否具有相同的好处存在不同的说法。这可能引发一个问题，即是否存在一个网络无法理解物理学的最小深度/宽度 [173]。PINN的互操作性在未来研究中也将发挥重要作用 [149]。对PINN的激活

function is needed. Jagtap et al [70] show that scalable activation function may be tuned to maximize network performance, in terms of convergence rate and solution correctness. Further research can look into alternative or hybrid methods of differentiating the differential equations. To speed up PINN training, the loss function in Chiu et al [33] is defined using numerical differentiation and automatic differentiation. The proposed can-PINN, i.e. coupled-automatic-numerical differentiation PINN, shows to be more sample efficient and more accurate than traditional PINN; because PINN with automatic differentiation can only achieve high accuracy with many collocation points. While the PINNs training points can be distributed spatially and temporally, making them highly versatile, on the other hand, the position of training locations affects the quality of the results. One downside of PINNs is that the boundary conditions must be established during the training stage, which means that if the boundary conditions change, a new network must be created [183].

As for the loss, it is important to note that a NN will always focus on minimizing the largest loss terms in the weighted equation, therefore all loss terms must be of the same order of magnitude; increasing emphasis on one component of the loss may affect other parts of it. There does not appear to be an objective method for determining the weight variables in the loss equation, nor does there appear to be a mechanism to assure that a certain equation can be solved to a predetermined tolerance before training begins; these are topics that still need to be researched [119].

It seems there has been a lack of research on optimization tasks for PINNs. Not many solutions appear to be implemented, apart from standard solutions such as Adam and BFGS algorithms [184]. The Adam algorithm generates a workflow that can be studied using dynamical systems theory, giving a gradient descent dynamic. More in detail, to reduce stiffness in gradient flow dynamics, studying the limiting neural tangent kernel is needed. Given that there has been great work to solve optimization problems or improve these approaches in machine learning, there is still room for improvement in PINNs optimization techniques [169]. The L-BFGS-B is the most common BFGS used in PINNs, and it is now the most critical PINN technology Markidis [105].
Moreover, the impact of learning rate on PINN training behavior has not been fully investigated. Finally, gradient normalization is another key research topic [120]. It is an approach that dynamically assigns weights to different constraints to remove the dominance of any component of the global loss function.

It is necessary to investigate an error estimation for PINN. One of the few examples comes from Hillebrecht and Unger [62], where using an ODE, they construct an upper bound on the PINN prediction error. They suggest adding an additional weighting parameter to the physics-inspired part of the loss function, which allows for balancing the error contribution of the initial condition and the ODE residual. Unfortunately, only a toy example is offered in this article, and a detailed analysis of the possibility of providing lower bounds on the error estimator needs still to be addressed, as well as an extension to PDEs.

5.3 PINN in the SciML Framework

PINNs, and SciML in general, hold a lot of potential for applying machine learning to critical scientific and technical challenges. However, many questions remain unsolved, particularly if neural networks are to be used as a replacement for traditional numerical methods such as finite difference or finite volume. In Krishnapriyan et al [86] the authors analyze two basic PDE problems of diffusion and convection and show that PINN can fail to learn the ph problem physics when convection or viscosity coefficients are high. They found that the

需要一种激活函数。Jagtap等人 [70] 表明，可扩展激活函数可以调整以最大化网络性能，包括收敛速度和解的正确性。进一步的研究可以探索微分方程的不同或混合微分方法。为了加速PINN训练，Chiu等人 [33] 使用数值微分和自动微分定义了损失函数。所提出的can-PINN，即耦合自动数值微分PINN，被证明比传统PINN更样本高效、更准确；因为使用自动微分的PINN只能通过许多配置点才能实现高精度。虽然PINNs的训练点可以在空间和时间上分布，使其非常通用，但另一方面，训练位置的位置会影响结果的质量。PINNs的一个缺点是边界条件必须在训练阶段建立，这意味着如果边界条件发生变化，必须创建一个新的网络 [183]。

关于损失，重要的是要注意神经网络（NN）始终专注于最小化加权方程中最大的损失项，因此所有损失项必须具有相同的数量级；对损失的一个分量增加重视可能会影响其其他部分。目前似乎没有客观的方法来确定损失方程中的权重变量，也没有机制来保证在训练开始之前某个方程可以以预定的容差求解；这些仍然是需要研究的问题 [119]。

似乎对物理信息神经网络（PINNs）的优化任务的研究还很不足。除了标准的Adam算法和BFGS算法之外，似乎没有实施太多解决方案 [184]。Adam算法生成一个工作流程，可以使用动力系统理论来研究，给出梯度下降动态。更详细地说，为了减少梯度流动态中的刚度，需要研究极限神经切线核。鉴于在机器学习中已经有很多工作来解决优化问题或改进这些方法，PINNs的优化技术仍有改进的空间 [169]。L-BFGS-B是PINNs中最常用的BFGS算法，现在它是马克迪斯（Markidis）最关键的PINN技术 [105]。

此外，学习率对PINN训练行为的影响尚未得到充分研究。最后，梯度归一化是另一个关键的研究课题 [120]。这是一种动态分配权重给不同约束的方法，以消除全局损失函数中任何组成部分的主导地位。

有必要研究PINN的误差估计。其中少数例子来自Hillebrecht和Unger [62], , 他们使用常微分方程构建了PINN预测误差的上界。他们建议在损失函数的物理启发部分添加一个额外的加权参数，这允许平衡初始条件和常微分方程残差的误差贡献。不幸的是，这篇文章只提供了一个玩具示例，并且需要进一步详细分析提供误差估计器下界的可能性，以及将其扩展到偏微分方程。

5.3 PINN在SciML框架中

PINNs以及SciML本身，在将机器学习应用于关键科学和技术挑战方面具有很大的潜力。然而，许多问题仍未解决，特别是当神经网络被用作传统数值方法（如有限差分法或有限体积法）的替代品时。在克里希纳普拉扬等人 [86] 中，作者分析了两个基本的偏微分方程问题：扩散和对流，并表明当对流或粘度系数较高时，PINN无法学习物理问题的物理规律。他们发现，

PINN loss landscape becomes increasingly complex for large coefficients. This is partly due to an optimization problem, because of the PINN soft constraint. However, lower errors are obtained when posing the problem as a sequence-to-sequence learning task instead of solving for the entire space-time at once. These kinds of challenges must be solved if PINN has to be used beyond basic copy-paste by creating in-depth relations between the scientific problem and the machine learning approaches.

Moreover, unexpected uses of PINN can result from applying this framework to different domains. PINN has been employed as linear solvers for the Poisson equation Markidis [105], by bringing attention to the prospect of using PINN as linear solvers that are as quick and accurate as other high-performance solvers such as PETSc solvers. PINNs appear to have some intriguing advantages over more traditional numerical techniques, such as the Finite Element Method (FEM), as explained in Lu et al [100]. Given that PINNs approximate functions and their derivatives nonlinearly, whereas FEM approximates functions linearly, PINNs appear to be well suited for broad use in a wide variety of engineering applications. However, one of the key disadvantages is the requirement to train the NN, which may take significantly longer time than more extensively used numerical methods.

On the other hand, PINNs appear to be useful in a paradigm distinct from that of standard numerical approaches. PINNs can be deployed in an online-offline fashion, with a single PINN being utilized for rapid evaluations of dynamics in real-time, improving predictions. Moving from 2D to 3D poses new obstacles for PINN. As training complexity grows in general, there is a requirement for better representation capacity of neural networks, a demand for a larger batch size that can be limited by GPU memory, and an increased training time to convergence [119]. Another task is to incorporate PINN into more traditional scientific programs and libraries written in Fortran and C/C++, as well as to integrate PINN solvers into legacy HPC applications [105]. PINN could also be implemented on Modern HPC Clusters, by using Horovod [156]. Additionally, when developing the mathematical model that a PINN will solve, the user should be aware of pre-normalizing the problem. At the same time, packages can assist users in dealing with such problems by writing the PDEs in a symbolic form, for example, using SymPy.

PINNs have trouble propagating information from the initial condition or boundary condition to unseen areas of the interior or to future times as an iterative solver [41, 73]. This aspect has recently been addressed by Wang et al [180] that provided a re-formulation of PINNs loss functions that may explicitly account for physical causation during model training. They assess that PINN training algorithms should be designed to respect how information propagates in accordance with the underlying rules that control the evolution of a given system. With the new implementation they observe considerable accuracy gains, as well as the possibility to assess the convergence of a PINNs model, and so PINN, can run for the chaotic Lorenz system, the Kuramoto–Sivashinsky equation in the chaotic domain, and the Navier–Stokes equations in the turbulent regime. However there is still research to be done for hybrid/inverse problems, where observational data should be considered as point sources of information, and PDE residuals should be minimized at those points before propagating information outwards. Another approach is to use ensemble agreement as to the criterion for incorporating new points in the loss calculated from PDEs [58]. The idea is that in the neighborhood of observed/initial data, all ensemble members converge to the same solution, whereas they may be driven towards different incorrect solutions further away from the observations, especially or large time intervals.

PINN can also have a significant impact on our daily lives, as for the example, from Yucesan and Viana [194], where PINNs are used to anticipate grease maintenance; in the industry 4.0 paradigm, they can assist engineers in simulating materials and constructions

当系数较大时，物理信息神经网络的损失景观会变得越来越复杂。这部分是由于优化问题，因为物理信息神经网络存在软约束。然而，当将问题作为序列到序列学习任务提出时，而不是一次性求解整个时空，可以获得更低的误差。如果物理信息神经网络要超越基本的复制粘贴，就必须通过在科学问题与机器学习方法之间建立深入关系来解决这类挑战。

此外，将此框架应用于不同领域可能会导致物理信息神经网络的意外用途。物理信息神经网络已被用作泊松方程的线性求解器马克迪斯 [105],，通过关注物理信息神经网络作为线性求解器的潜力，这些求解器与其他高性能求解器（如PETSc求解器）一样快速和准确。正如陆等人 [100] 所解释的那样，物理信息神经网络与传统数值技术（如有限元方法（FEM））相比，似乎具有一些引人入胜的优势。鉴于物理信息神经网络非线性地近似函数及其导数，而有限元方法线性地近似函数，物理信息神经网络似乎非常适合在广泛的工程应用中广泛使用。然而，其主要缺点之一是训练神经网络的要求，这可能比更广泛使用的数值方法需要显著更长的时间。

另一方面，物理信息神经网络（PINNs）似乎在不同于标准数值方法的范式中有用。PINNs可以采用在线-离线方式部署，单个PINN可用于实时快速评估动力学，从而提高预测。从2D到3D为PINN带来了新的挑战。随着训练复杂性的普遍增长，对神经网络表示能力的要求更高，需要更大的批处理大小（这可能受GPU内存限制），以及增加收敛所需的训练时间 [119]。另一项任务是，将PINN集成到用Fortran和C/C++编写的更传统的科学程序和库中，以及将PINN求解器集成到遗留HPC应用中 [105]。PINN也可以在现代HPC集群上实现，例如使用Horovod [156]。此外，在开发PINN将求解的数学模型时，用户应注意对问题进行预归一化。同时，软件包可以通过以符号形式编写偏微分方程（PDEs），例如使用SymPy，来帮助用户处理这类问题。

物理信息神经网络（PINNs）在将信息从初始条件或边界条件传播到内部未见过区域或未来时间方面存在困难，这就像一个迭代求解器 [41, 73]。最近，王等人 [180] 解决了这个问题，他们提供了一种重新表述的PINNs损失函数，可以在模型训练过程中明确考虑物理因果关系。他们认为，PINN训练算法应该设计为尊重信息传播的规则，这些规则控制着给定系统的演化。通过新的实现，他们观察到显著的精度提升，以及评估PINNs模型收敛性的可能性，因此PINN可以运行在混沌的洛伦兹系统、混沌域中的Kuramoto–Sivashinsky方程以及湍流状态中的纳维-斯托克斯方程。然而，对于混合/逆问题，仍需进行研究，在这些问题中，观测数据应被视为信息点源，并且在向外传播信息之前，应在这些点处最小化PDE残差。另一种方法是使用集成一致性作为将新点纳入从PDEs计算的损失中的标准 [58]。其思想是，在观测/初始数据的邻域内，所有集成成员都收敛到相同的解，而在远离观测点的区域，它们可能会被驱动到不同的错误解，尤其是在较大的时间间隔内。

物理信息神经网络（PINN）也能对我们的日常生活产生重大影响，例如，在余和Viana [194], 的研究中，物理信息神经网络被用于预测润滑维护；在工业4.0范式中，它们可以协助工程师模拟材料和结构

or analyzing in real-time buildings structures by embedding elastostatic trained PINNs [57, 110]. PINNs also fail to solve PDEs with high-frequency or multi-scale structure [47, 179, 181]. The region of attraction of a specific equilibria of a given autonomous dynamical system could also be investigated with PINN [152].

However, to employ PINN in a safety-critical scenario it will still be important to analyze stability and focus on the method's more theoretical component. Many application areas still require significant work, such as the cultural heritage sector, the healthcare sector, fluid dynamics, particle physics, and the modeling of general relativity with PINN.

It will be important to develop a PINN methodology for stiff problems, as well as use PINN in digital twin applications such as real-time control, cybersecurity, and machine health monitoring [119]. Finally, there is currently a lack of PINNs applications in multi-scale applications, particularly in climate modeling [68], although the PINN methodology has proven capable of addressing its capabilities in numerous applications such as bubble dynamics on multiple scales [95, 96].

5.4 PINN in the AI Framework

PINN could be viewed as a building block in a larger AI framework, or other AI technologies could help to improve the PINN framework.

For more practical applications, PINNs can be used as a tool for engaging deep reinforcement learning (DRL) that combines reinforcement Learning (RL) and deep learning. RL enables agents to conduct experiments to comprehend their environment better, allowing them to acquire high-level causal links and reasoning about causes and effects [11]. The main principle of reinforcement learning is to have an agent learn from its surroundings through exploration and by defining a reward [159]. In the DRL framework, the PINNs can be used as agents. In this scenario, information from the environment could be directly embedded in the agent using knowledge from actuators, sensors, and the prior-physical law, like in a transfer learning paradigm.

PINNs can also be viewed as an example of merging deep learning with symbolic artificial intelligence. The symbolic paradigm is based on the fact that intelligence arises by manipulating abstract models of representations and interactions. This approach has the advantage to discover features of a problem using logical inference, but it lacks the easiness of adhering to real-world data, as in DL. A fulfilling combination of symbolic intelligence and DL would provide the best of both worlds. The model representations could be built up directly from a small amount of data with some priors [50]. In the PINN framework, the physical injection of physical laws could be treated as symbolic intelligence by adding reasoning procedures.

Causal Models are intermediate descriptions that abstract physical models while answering statistical model questions [154]. Differential equations model allows to forecast a physical system's future behavior, assess the impact of interventions, and predict statistical dependencies between variables. On the other hand, a statistical model often doesn't refer to dynamic processes, but rather how some variables allow the prediction of others when the experimental conditions remain constant. In this context, a causal representation learning, merging Machine learning and graphical causality, is a novel research topic, and given the need to model physical phenomena given known data, it may be interesting to investigate whether

或通过嵌入弹性静力学训练的PINNs [57,110] 实时分析建筑物结构。PINNs也无法解决具有高频或多尺度结构的偏微分方程 [47, 179,181]。还可以使用PINN [152] 研究给定自洽动力系统的特定平衡点的吸引域。

然而，要在安全关键场景中使用PINN，仍然需要分析稳定性并关注该方法的理论成分。许多应用领域仍需大量工作，例如文化遗产部门、医疗保健部门、流体动力学、粒子物理学，以及使用PINN进行广义相对论建模。

开发用于刚性问题PINN方法以及将PINN用于数字孪生应用（如实时控制、网络安全和机器健康监测） [119] 将非常重要。最后，目前多尺度应用中PINNs应用仍然缺乏，特别是在气候建模 [68]，尽管PINN方法已证明在众多应用（如多尺度气泡动力学） [95, 96] 中具备其能力。

5.4 物理信息神经网络在人工智能框架中

物理信息神经网络可以被看作是更广泛的人工智能框架中的一个构建块，或者其他人工智能技术可以帮助改进物理信息神经网络框架。

对于更实际的应用，物理信息神经网络可以作为一种工具，用于结合强化学习（RL）和深度学习的深度强化学习（DRL）。强化学习使智能体能够通过实验来更好地理解其环境，从而获得高级因果关系并进行原因和结果的推理 [11]。强化学习的主要原则是让智能体通过探索和定义奖励来从其周围环境中学习 [159]。在深度强化学习框架中，物理信息神经网络可以作为智能体使用。在这种情况下，来自环境的信息可以直接使用来自执行器、传感器和先验物理定律的知识嵌入智能体，就像在迁移学习范例中一样。

物理信息神经网络也可以被看作是深度学习与符号人工智能融合的一个例子。符号范式基于这样一个事实：智能通过操作表示和交互的抽象模型而产生。这种方法的优势在于能够使用逻辑推理发现问题的特征，但它缺乏遵循真实世界数据的便利性，就像在深度学习（DL）中一样。符号智能和深度学习的完美结合将提供两者的最佳优势。模型表示可以直接从少量数据和一些先验知识 [50] 构建起来。在物理信息神经网络框架中，物理定律的注入可以被看作是符号智能，通过添加推理程序来实现。

因果模型是介于物理模型和数据模型之间的中间描述，同时回答数据模型问题 [154]。微分方程模型允许预测物理系统的未来行为、评估干预措施的影响，并预测变量之间的统计依赖关系。另一方面，数据模型通常不涉及动态过程，而是描述在实验条件保持不变时，某些变量如何允许预测其他变量。在这种情况下，因果表示学习，融合机器学习和图形因果性，是一个新兴的研究课题，鉴于在已知数据的情况下需要模拟物理现象，研究它是否可能是有趣的。

PINN, can help to determine causality when used to solve hybrid (mixed forward and inverse) problems.

6 Conclusion

This review can be considered an in-depth study of an innovation process over the last four years rather than a simple research survey in the field of PINNs. Raissi's first research [143, 144], which developed the PINN framework, focused on implementing a PINN to solve known physical models. These innovative papers helped PINN methodology gain traction and justify its original concept even more. Most of the analyzed studies have attempted to personalize the PINNs by modifying the activation functions, gradient optimization procedures, neural networks, or loss function structures. A border extension of PINNs original idea brings to use in the physical loss function bare minimum information of the model, without using a typical PDE equation, and on the other side to embed directly in the NN structure the validity of initial or boundary conditions. Only a few have looked into alternatives to automatic differentiation [44] or at convergence problems [179, 181]. Finally, a core subset of publications has attempted to take this process to a new meta-level by proposing all-inclusive frameworks for many sub-types of physical problems or multi-physics systems [21]. The brilliance of the first PINN articles [143, 144] lies in resurrecting the concept of optimizing a problem with a physical constraint by approximating the unknown function with a neural network [39] and then extending this concept to a hybrid data-equation driven approach within modern research. Countless studies in previous years have approximated the unknown function using ways different than neural networks, such as the kernel approaches [126], or other approaches that have used PDE functions as constraints in an optimization problem [63].

However, PINNs are intrinsically driven by physical information, either from the data point values or the physical equation. The former can be provided at any point in the domain but is usually only as initial or boundary data. More importantly, the latter are the collocation points where the NN is forced to obey the physical model equation.

We examined the literature on PINNs in this paper, beginning with the first papers from Raissi et al [143, 144] and continuing with the research on including physical priors on Neural Networks. This survey looks at PINNs, as a collocation-based method for solving differential questions with Neural networks. Apart from the vanilla PINN solution we look at most of its variants, like variational PINN (VPINN) as well as in its soft form, with loss including the initial and boundary conditions, and its hard form version with boundary conditions encoded in the Neural network structure.

This survey explains the PINN pipeline, analyzing each building block; first, the neural networks, then the loss construction based on the physical model and feedback mechanism. Then, an overall analysis of examples of equations in which the PINN methodology has been used, and finally, an insight into PINNs on where they can be concretely applied and the packages available.

Finally, we can conclude that numerous improvements are still possible; most notably, in unsolved theoretical issues. There is still potential for development in training PINNs optimally and extending PINNs to solve multiple equations.

Funding Open access funding provided by Universitá degli Studi di Napoli Federico II within the CRUI-CARE Agreement. The authors have not disclosed any funding.

Data Availability Enquiries about data availability should be directed to the authors.

物理信息神经网络 (PINN)，在用于解决混合（正向和反向）问题时，可以帮助确定因果关系。

6 结论

这篇综述可以被认为是对过去四年创新过程的深入研究，而不是对物理信息神经网络 (PINN) 领域的简单研究调查。拉西的第一项研究 [143,144]，开发了物理信息神经网络框架，专注于实现一个物理信息神经网络来求解已知的物理模型。这些创新性论文帮助物理信息神经网络方法获得了关注，并进一步证明了其原始概念。大多数分析的研究都试图通过修改激活函数、梯度优化过程、神经网络或损失函数结构来个性化物理信息神经网络。物理信息神经网络原始想法的边界扩展带来了使用物理损失函数的最少模型信息，而无需使用典型的偏微分方程，另一方面则将初始或边界条件的有效性直接嵌入到神经网络结构中。只有少数研究探讨了自动微分的替代方案 [44] 或收敛性问题 [179, 181]。最后，一小部分出版物试图通过提出适用于许多物理问题子类型或多物理系统的全面框架 [21] 来将这一过程提升到新的元层面。最初的物理信息神经网络文章 [143, 144] 的卓越之处在于，通过用神经网络逼近未知函数 [39] 来优化具有物理约束的问题，并将这一概念扩展到现代研究中的混合数据-方程驱动方法。过去多年的无数研究使用不同于神经网络的方式逼近未知函数，例如核方法 [126]，或在其他方法中使用偏微分方程函数作为优化问题的约束 [63]。

然而，PINNs本质上是由物理信息驱动的，这些信息可以来自数据点值或物理方程。前者可以在域中的任何点提供，但通常仅作为初始或边界数据。更重要的是，后者是配置点，在这些点上，神经网络被强制遵守物理模型方程。

我们在本文中研究了PINNs的文献，从Raissi等人的第一篇论文 [143, 144] 开始，继续研究了在神经网络中包含物理先验的研究。这项调查将PINNs视为一种基于配置的方法，用于使用神经网络解决微分问题。除了基础物理信息神经网络解法外，我们还研究了其大多数变体，如变分物理信息神经网络（VPINN）以及其软形式，其中损失包括初始和边界条件，以及其硬形式版本，其中边界条件编码在神经网络结构中。

这项调查解释了PINN流程，分析了每个构建块；首先，神经网络，然后基于物理模型和反馈机制的损失构建。接着，对PINN方法已使用的方程示例进行整体分析，最后，对PINNs在具体应用领域以及可用软件包的见解。

最后，我们可以得出结论，仍然有许多改进空间；最重要的是，在未解决的理论问题上。在优化训练PINNs和将PINNs扩展到解决多个方程方面，仍有发展潜力。

资助由那不勒斯费德里科二世大学在CRUI-CARE协议内提供的开放获取资助。作者未披露任何资助。

数据可用性关于数据可用性的查询应直接联系作者。

Declarations

Competing interests The authors have not disclosed any competing interests.

References

1. Abreu, E., Florindo, J.B.: A Study on a Feedforward Neural Network to Solve Partial Differential Equations in Hyperbolic-Transport Problems. In: Paszynski M, Kranzlmüller D, Krzhizhanovskaya VV, et al (eds) Computational Science – ICCS 2021. Springer International Publishing, Cham, Lecture Notes in Comput. Sci. pp. 398–411, (2021) https://doi.org/10.1007/978-3-030-77964-1_31
2. Aldweesh, A., Derhab, A., Emam, A.Z.: Deep learning approaches for anomaly-based intrusion detection systems: A survey, taxonomy, and open issues. Knowledge-Based Systems **189**, 105,124 (2020). https://doi.org/10.1016/j.knosys.2019.105124, https://www.sciencedirect.com/science/article/pii/S0950705119304897
3. Alkhadhr, S., Liu, X., Almekkawy, M: Modeling of the Forward Wave Propagation Using Physics-Informed Neural Networks. In: 2021 IEEE International Ultrasonics Symposium (IUS), pp. 1–4, (2021) https://doi.org/10.1109/IUS52206.2021.9593574, iSSN: 1948-5727
4. Almajid, M.M., Abu-Al-Saud, M.O.: Prediction of porous media fluid flow using physics informed neural networks. J. Pet. Sci, Eng. **208**, 109,205 (2022). https://doi.org/10.1016/j.petrol.2021.109205, https://www.sciencedirect.com/science/article/pii/S0920410521008597
5. Alom, M.Z., Taha, T.M., Yakopcic, C., et al: A state-of-the-art survey on deep learning theory and architectures. Electron. **8**(3) (2019). https://doi.org/10.3390/electronics8030292, https://www.mdpi.com/2079-9292/8/3/292
6. Amini Niaki, S., Haghighat, E., Campbell, T., et al.: Physics-informed neural network for modelling the thermochemical curing process of composite-tool systems during manufacture. Computer Methods in Applied Mechanics and Engineering **384**, 113,959 (2021). https://doi.org/10.1016/j.cma.2021.113959, https://www.sciencedirect.com/science/article/pii/S0045782521002966
7. Araz, J.Y., Criado, J.C., Spannowsky, M: Elvet – a neural network-based differential equation and variational problem solver (2021). arXiv:2103.14575 [hep-lat, physics:hep-ph, physics:hep-th, stat] , arXiv: 2103.14575
8. Arnold, D.N.: Stability, Consistency, and Convergence of Numerical Discretizations, pp. 1358–1364. Springer, Berlin, Heidelberg (2015). https://doi.org/10.1007/978-3-540-70529-1_407
9. Arnold, F., King, R: State–space modeling for control based on physics-informed neural networks. Eng. Appl. Artif. Intell. **101**, 104,195 . https://doi.org/10.1016/j.engappai.2021.104195, https://www.sciencedirect.com/science/article/pii/S0952197621000427
10. Arthurs, C.J., King, A.P.: Active training of physics-informed neural networks to aggregate and interpolate parametric solutions to the Navier-Stokes equations. J. Comput. Phys. 438:110,364 (2021). https://doi.org/10.1016/j.jcp.2021.110364, https://www.sciencedirect.com/science/article/pii/S002199912100259X
11. Arulkumaran, K., Deisenroth, M.P., Brundage, M., et al.: Deep Reinforcement Learning: A Brief Survey. IEEE Signal Process. Mag. **34**(6), 26–38 (2017). https://doi.org/10.1109/MSP.2017.2743240
12. Balu, A., Botelho, S., Khara, B., et al.: Distributed multigrid neural solvers on megavoxel domains. In: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. Association for Computing Machinery, New York, NY, USA, SC '21, (2021) https://doi.org/10.1145/3458817.3476218
13. Bauer, B.: Kohler, M: On deep learning as a remedy for the curse of dimensionality in nonparametric regression. Ann. Statist. **47**(4), 2261–2285 (2019). https://doi.org/10.1214/18-

声明

利益冲突 作者未披露任何利益冲突。

参考文献

1. Abreu, E., Florindo, J.B.: 综述 基于前馈神经网络求解双曲传输问题中的偏微分方程。见于：Paszynski M, Kranzlmüller D, Krzhizhanovskaya VV, 等编《计算科学——2021年国际计算科学会议》。斯普林格国际出版公司, 查姆, 计算机科学讲义, pp. 398–411, (2021) https://doi.org/10.1007/978-3-030-77964-1_31.
2 Aldweesh, A., Derhab, A., Emam, A.Z.: 深度学习在基于异常的入侵检测系统中的应用：综述、分类和开放问题。知识系统 **189**,105,124(2020). https://doi.org/10.1016/j.knosys.2019.105124,https://www.sciencedirect.com/science/article/pii/S0950705119304897
3. Alkhadhr, S., Liu, X., Almekkawy, M: 基于物理信息神经网络的正向波传播建模。见于：2021年IEEE国际超声 symposium (IUS), pp. 1–4,(2021)https://doi.org/10.1109/IUS52206.2021.9593574,iSSN: 1948-5727
4. Almajid, M.M., Abu-Al-Saud, M.O.: 基于物理信息神经网络的 多孔介质 流体流动预测。石油科学和工程杂志 **208**,109,205 (2022)。 https://doi.org/10.1016/j.petrol.2021.109205,https://www.sciencedirect.com/science/article/pii/S0920410521008597
5. Alom, M.Z., Taha, T.M., Yakopcic, C., 等: 深度学习理论和架构的最新综述。电子杂志 **8**(3) (2019)。 https://doi.org/10.3390/electronics8030292, https://www.mdpi.com/2079-9292/8/3/292
6. 阿米尼·尼亚基, S., 哈吉加特, E., 坎贝尔,T., 等：物理信息神经网络用于建模制造过程中复合工具系统的热化学固化过程。应用力学与计算机方法**384**, 113,959 (2021)。 https://doi.org/10.1016/j.cma.2021.113959,https://www.sciencedirect.com/science/article/pii/S0045782521002966
7. 阿拉兹, J.Y., 克里亚多, J.C., 斯帕诺夫斯基, M: Elvet – 基于神经网络的微分方程和变分问题求解器 (2021)。arXiv:2103.14575 [hep-lat, physics:hep-ph, physics:hep-th, stat] ,arXiv: 2103.14575
8. 阿诺德, D.N.: 数值离散化的稳定性、一致性和收敛性，第 1358–1364 页。斯普林格, 柏林, 海德堡 (2015)。 https://doi.org/10.1007/978-3-540-70529-1_407
9. 阿诺德，F.，金，R: 基于物理信息神经网络的控制状态空间建模。工程应用人工智能**101**， 104,195 . https://doi.org/10.1016/j.engappai.2021.104195,https://www.sciencedirect.com/science/article/pii/S0952197621000427
10. Arthurs，C.J.，金，A.P.：用于纳维-斯托克斯方程参数解的聚合和插值的物理信息神经网络的主动训练。计算物理杂志 438:110,364 (2021)。 https://doi.org/10.1016/j.jcp.2021.110364,https://www.sciencedirect.com/science/article/pii/S002199912100259X
11. 阿鲁库马兰，K.，迪森罗特，M.P.，布兰达奇，等：深度强化学习：简要综述。IEEE 信号处理杂志**34**(6)， 26–38 (2017)。 https://doi.org/10.1109/MSP.2017.2743240
12. 巴拉，A.，博特洛，S.，卡拉，B.，等：分布式多重网格神经网络求解器在兆体素域上的应用。在：高性能计算、网络、存储和分析国际会议论文集。美国纽约，纽约州，美国，ACM SC’21, (2021) https://doi.org/10.1145/3458817.3476218
13. 鲍尔,B.: 科尔勒，M: 深度学习作为非参数回归中维数灾难的补救措施。统计年鉴 **47**(4)，2261–2285 (2019)。 https://doi.org/10.1214/18-

AOS1747, http://projecteuclid.org/journals/annals-of-statistics/volume-47/issue-4/On-deep-learning-as-a-remedy-for-the-curse-of/10.1214/18-AOS1747.full
14. Belkin, M., Hsu, D., Ma, S., et al.: Reconciling modern machine-learning practice and the classical bias-variance trade-off. Proc. Nat. Acad. Sci. India Sect. **116**(32), 15849–15854 (2019). https://doi.org/10.1073/pnas.1903070116, www.pnas.org/doi/10.1073/pnas.1903070116
15. Bellman, R.: Dynamic programming. Sci. **153**(3731), 34–37 (1966)
16. Berg, J., Nyström, K.: A unified deep artificial neural network approach to partial differential equations in complex geometries. Neurocomputing **317**, 28–41 (2018). https://doi.org/10.1016/j.neucom.2018.06.056, www.sciencedirect.com/science/article/pii/S092523121830794X
17. Berman, D.S., Buczak, A.L., Chavis, J.S., et al.: A survey of deep learning methods for cyber security. Inform. **10**(4) (2019). https://doi.org/10.3390/info10040122, https://www.mdpi.com/2078-2489/10/4/122
18. Biswas, A., Tian, J., Ulusoy, S.: Error estimates for deep learning methods in fluid dynamics. Numer. Math. **151**(3), 753–777 (2022). https://doi.org/10.1007/s00211-022-01294-z
19. Blechschmidt, J., Ernst, O.G.: Three ways to solve partial differential equations with neural networks – A review. GAMM-Mitteilungen **44**(2), e202100,006 (2021). https://doi.org/10.1002/gamm.202100006, https://onlinelibrary.wiley.com/doi/abs/10.1002/gamm.202100006
20. Cai, S., Mao, Z., Wang, Z., et al.: Physics-informed neural networks (PINNs) for fluid mechanics: a review. Acta. Mech. Sin. **37**(12), 1727–1738 (2021). https://doi.org/10.1007/s10409-021-01148-1
21. Cai, S., Wang, Z., Lu, L., et al.: DeepM&Mnet: Inferring the electroconvection multiphysics fields based on operator approximation by neural networks. J. Comput. Phys. **436**, 110,296 (2021b). https://doi.org/10.1016/j.jcp.2021.110296, https://www.sciencedirect.com/science/article/pii/S0021999121001911
22. Cai, S., Wang, Z., Wang, S., et al.: Physics-Informed Neural Networks for Heat Transfer Problems. J. Heat Transf. **143**(6) (2021c). https://doi.org/10.1115/1.4050542
23. Calin, O.: Convolutional Networks, pp. 517–542. Springer International Publishing, Cham (2020). https://doi.org/10.1007/978-3-030-36721-3_16
24. Calin, O.: Universal Approximators, pp. 251–284. Springer Series in the Data Sciences, Springer International Publishing, Cham (2020b). https://doi.org/10.1007/978-3-030-36721-3_9
25. Caterini, A.L., Chang, D.E.: In: Caterini, A.L., Chang, D.E. (eds.) Deep Neural Networks in a Mathematical Framework. Generic Representation of Neural Networks, pp. 23–34. SpringerBriefs in Computer Science, Springer International Publishing, Cham (2018). https://doi.org/10.1007/978-3-319-75304-1_3
26. Caterini, A.L., Chang, D.E.: Specific Network Descriptions. In: Caterini, A.L., Chang, D.E. (eds.) Deep Neural Networks in a Mathematical Framework, pp. 35–58. Springer International Publishing, Cham, SpringerBriefs in Computer Science (2018). https://doi.org/10.1007/978-3-319-75304-1_4
27. Cavanagh, H., Mosbach, A., Scalliet, G., et al.: Physics-informed deep learning characterizes morphodynamics of asian soybean rust disease. Nat. Commun. **12**(1), 6424 (2021). https://doi.org/10.1038/s41467-021-26577-1
28. Chen, F., Sondak, D., Protopapas, P., et al.: Neurodiffeq: A python package for solving differential equations with neural networks. J. Open Source Softw. **5**(46), 1931 (2020)
29. Chen, H., Engkvist, O., Wang, Y., et al.: The rise of deep learning in drug discovery. Drug Discov. Today **23**(6), 1241–1250 (2018). https://doi.org/10.1016/j.drudis.2018.01.039, www.sciencedirect.com/science/article/pii/S1359644617303598
30. Chen, Y., Lu, L., Karniadakis, G.E., et al.: Physics-informed neural networks for inverse problems in nano-optics and metamaterials. Opt. Express **28**(8), 11618–11633 (2020). https://doi.org/10.1364/OE.384875, www.osapublishing.org/oe/abstract.cfm?uri=oe-28-8-11618
31. Cheng, C., Zhang, G.T.: Deep Learning Method Based on Physics Informed Neural Network with Resnet Block for Solving Fluid Flow Problems. Water **13**(4), 423 (2021). https://doi.org/10.3390/w13040423, www.mdpi.com/2073-4441/13/4/423
32. Cheung, K.C., See, S.: Recent advance in machine learning for partial differential equation. CCF Trans. High Performance Comput. **3**(3), 298–310 (2021). https://doi.org/10.1007/s42514-021-00076-7
33. Chiu, P.H., Wong, J.C., Ooi, C., et al.: CAN-PINN: A fast physics-informed neural network based on coupled-automatic–numerical differentiation method. Comput. Methods Appl. Mech. Engrg. **395**, 114,909 (2022). https://doi.org/10.1016/j.cma.2022.114909, https://www.sciencedirect.com/science/article/pii/S0045782522001906
34. Cybenko, G.: Approximation by superpositions of a sigmoidal function. Math. Control Signals Systems **2**(4), 303–314 (1989). https://doi.org/10.1007/BF02551274
35. Dargan, S., Kumar, M., Ayyagari, M.R., et al.: A Survey of Deep Learning and Its Applications: A New Paradigm to Machine Learning. Arch. Comput. Methods Engrg. **27**(4), 1071–1092 (2020). https://doi.org/10.1007/s11831-019-09344-w

AOS1747, , http://projecteuclid.org/journals/annals-of-statistics/volume-47/issue-4/On-deep-learning-as-a-remedy-for-the-curse-of/10.1214/18-AOS1747.full
14. 贝尔金，M.，许，D.，马，S.，等：协调现代机器学习实践与经典偏差-方差权衡。印度国家科学院公报第 **116**(32), 15849–15854(2019)。 https://doi.org/10.1073/pnas.1903070116， www.pnas.org/doi/10.1073/pnas.1903070116Be
15. llman, R.: 动态规划。科学 **153**(3731)，34–37 (1966)
16. 伯格，J.，尼斯特罗姆，K.：一种统一的深度人工神经网络方法用于复杂几何形状中的偏微分方程。神经计算 **317**，28–41 (2018)。 https://doi.org/10.1016/j.neucom.2018.06.056， www.sciencedirect.com/science/article/pii/S092523121830794X
17. 伯曼，D.S.，布查克，A.L.，查维斯，J.S.，等：用于网络安全的深度学习方法综述。Inform. **10**(4) (2019). https://doi.org/10.3390/info10040122,https://www.mdpi.com/2078-2489/10/4/122
18. 比斯瓦斯，A.，田，J.，乌卢索伊，S.：流体动力学中深度学习方法的误差估计。数值数学 **151**(3)，753–777 (2022).https://doi.org/10.1007/s00211-022-01294-z
19. 布莱希施密特，J.，厄恩斯特，O.G.：用神经网络求解偏微分方程的三种方法——综述。GAMM简报 **44**(2), e202100,006(2021). https://doi.org/10.1002/gamm.202100006,https://onlinelibrary.wiley.com/doi/abs/10.1002/gamm.202100006
20. 蔡，S.，毛，Z.，王，Z.，等：用于流体力学的物理信息神经网络（PINNs）：综述。力学学报 **37**(12)，1727–1738(2021)。 https://doi.org/10.1007/s10409-021-01148-1
21. 蔡，S.，王，Z.，陆，L.，等：DeepM&Mnet：基于神经网络算子逼近的电对流多物理场推断。计算物理杂志 **436**，110,296 (2021b)。 https://doi.org/10.1016/j.jcp.2021.110296， https://www.sciencedirect.com/science/article/pii/S0021999121001911
22. 蔡，S.，王，Z.，王，S.，等：用于热传递问题的物理信息神经网络。传热杂志 **143**(6) (2021c)。https://doi.org/10.1115/1.4050542
23. 卡林，O.：卷积网络，第517–542页。斯普林格国际出版公司，查姆（2020）。https://doi.org/10.1007/978-3-030-36721-3_16
24. 卡林，O.：通用逼近器，第251–284页。斯普林格数据科学系列，斯普林格国际出版公司，查姆（2020b）。 https://doi.org/10.1007/978-3-030-36721-3_9
25. 卡特里尼，A.L.，张，D.E.：见于卡特里尼，A.L.，张，D.E.（编）深度神经网络在数学框架中。神经网络的通用表示，第23–34页。斯普林格计算机科学简报，斯普林格国际出版公司，查姆（2018）。https://doi.org/10.1007/978-3-319-75304-1_3
26. 卡特里尼，A.L.，张，D.E.：特定网络描述。见于卡特里尼，A.L.，张，D.E.（编）深度神经网络在数学框架中，第35–58页。斯普林格国际出版公司，查姆，斯普林格计算机科学简报（2018）。https://doi.org/10.1007/978-3-319-75304-1_4
27. 卡万纳，H.，莫斯巴赫，A.，斯卡利特，G.，等：物理信息深度学习表征亚洲大豆锈病的形态动力学。自然通信**12**(1)，6424 (2021)。https://doi.org/10.1038/s41467-021-26577-1
28. 陈，F.，桑达克，D.，普罗托帕帕斯，P.，等：Neurodiffeq：一个用于用神经网络求解微分方程的Python包。开源软件杂志<样式id='1'>5</样式>(46)，1931(2020)
29. 陈，H.，恩格奎斯特，O.，王，Y.，等：深度学习在药物发现中的兴起。药物发现今日<样式id='3'>23</样式><样式id='5'>(6), 1241–1250 (2018)。</样式><样式id='7'>https://doi.org/10.1016/j.drudis.2018.01.039</样式><样式id='9'>，</样式><样式id='11'>www.sciencedirect.com/ science/article/pii/S1359644617303598</样式>
30. 陈，Y.，陆，L.，Karniadakis，G.E.，等：物理信息神经网络用于纳米光学和超材料中的逆问题。光表达**28**(8), 11618–11633(2020)。https://doi.org/10.1364/OE.384875， www.osapublishing.org/oe/abstract.cfm?uri=oe-28-8-11618
31. 程,C.，张，G.T.：基于物理信息神经网络的深度学习方法，结合Resnet模块用于解决流体流动问题。水**13**(4)，423 (2021)。https://doi.org/10.3390/w13040423， www.mdpi.com/2073-4441/13/4/423
32. 张，K.C.，徐，S.：机器学习在偏微分方程中的最新进展。CCFTrans.高性能计算**3**(3)，298–310 (2021)。 https://doi.org/10.1007/s42514-021-00076-7
33. 周, P.H。，王, J.C。，黄, C。，等：CAN-PINN：基于耦合自动数值微分方法的快速物理信息神经网络。计算方法与应用力学**395**，114,909 (2022年)。 https://doi.org/10.1016/j.cma.2022.114909,https://www.sciencedirect.com/science/article/pii/S0045782522001906
34. 赛本科,G：sigmoidal函数叠加的逼近。数学控制信号系统**2**(4)，303–314 (1989年)。 https://doi.org/10.1007/BF02551274
35. 达尔根, S。，库马尔, M。，阿亚加里,M.R。，等：深度学习及其应用综述：机器学习的新范式。工程计算方法档案**27**(4)，1071–1092 (2020年)。 https://doi.org/10.1007/s11831-019-09344-w

36. De Ryck, T., Mishra, S.: Error analysis for physics informed neural networks (PINNs) approximating Kolmogorov PDEs. (2021) arXiv:2106.14473 [cs, math]
37. De Ryck, T., Lanthaler, S., Mishra, S.: On the approximation of functions by tanh neural networks. Neural Netw. **143**, 732–750 (2021). https://doi.org/10.1016/j.neunet.2021.08.015, www.sciencedirect.com/science/article/pii/S0893608021003208
38. De Ryck, T., Jagtap, A.D., Mishra, S.: Error estimates for physics informed neural networks approximating the Navier-Stokes equations. (2022) arXiv:2203.09346 [cs, math]
39. Dissanayake, M.W.M.G., Phan-Thien, N.: Neural-network-based approximations for solving partial differential equations. Commun. Numer. Methods Eng. **10**(3), 195–201 (1994). https://doi.org/10.1002/cnm.1640100303, https://onlinelibrary.wiley.com/doi/abs/10.1002/cnm.1640100303
40. Driscoll, T.A., Hale, N., Trefethen, L.N.: Chebfun Guide. Pafnuty Publications. http://www.chebfun.org/docs/guide/ (2014)
41. Dwivedi, V., Srinivasan, B.: Physics Informed Extreme Learning Machine (PIELM)-A rapid method for the numerical solution of partial differential equations. Neurocomputing **391**, 96–118 (2020). https://doi.org/10.1016/j.neucom.2019.12.099, www.sciencedirect.com/science/article/pii/S0925231219318144
42. EW, Yu. B.: The Deep Ritz Method: A Deep Learning-Based Numerical Algorithm for Solving Variational Problems. Commun. Math. Stat. **6**(1), 1–12 (2018). https://doi.org/10.1007/s40304-018-0127-z
43. Elbrächter, D., Perekrestenko, D., Grohs, P., et al.: Deep Neural Network Approximation Theory. IEEE Trans. Inf. Theory **67**(5), 2581–2623 (2021). https://doi.org/10.1109/TIT.2021.3062161
44. Fang, Z.: A High-Efficient Hybrid Physics-Informed Neural Networks Based on Convolutional Neural Network. IEEE Transactions on Neural Networks and Learning Systems pp. 1–13. (2021) https://doi.org/10.1109/TNNLS.2021.3070878
45. Fang, Z., Zhan, J.: Deep Physical Informed Neural Networks for Metamaterial Design. IEEE Access **8**, 24506–24513 (2020). https://doi.org/10.1109/ACCESS.2019.2963375
46. Fang, Z., Zhan, J.: A Physics-Informed Neural Network Framework for PDEs on 3D Surfaces: Time Independent Problems. IEEE Access **8**, 26328–26335 (2020). https://doi.org/10.1109/ACCESS.2019.2963390
47. Fuks, O., Tchelepi, H.A.: LIMITATIONS OF PHYSICS INFORMED MACHINE LEARNING FOR NONLINEAR TWO-PHASE TRANSPORT IN POROUS MEDIA. Journal of Machine Learning for Modeling and Computing **1**(1) (2020). https://doi.org/10.1615/.2020033905, https://www.dl.begellhouse.com/journals/558048804a15188a,583c4e56625ba94e,415f83b5707fde65.html
48. Gao, H., Sun, L., Wang, J.X.: PhyGeoNet: Physics-informed geometry-adaptive convolutional neural networks for solving parameterized steady-state PDEs on irregular domain. J. Comput. Phys. **428**, 110,079 (2021). https://doi.org/10.1016/j.jcp.2020.110079, https://www.sciencedirect.com/science/article/pii/S0021999120308536
49. Gardner, J.R., Pleiss, G., Bindel, D., et al.: Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration. In: Advances in Neural Information Processing Systems (2018)
50. Garnelo, M., Shanahan, M.: Reconciling deep learning with symbolic artificial intelligence: representing objects and relations. Curr. Opinion in Behav. Sci. **29**, 17–23 (2019). https://doi.org/10.1016/j.cobeha.2018.12.010, www.sciencedirect.com/science/article/pii/S2352154618301943
51. Geneva, N., Zabaras, N.: Modeling the dynamics of pde systems with physics-constrained deep auto-regressive networks. J. Comput. Phys. **403**, 109,056 (2020). https://doi.org/10.1016/j.jcp.2019.109056, https://www.sciencedirect.com/science/article/pii/S0021999119307612
52. Goswami, S., Anitescu, C., Chakraborty, S., et al.: Transfer learning enhanced physics informed neural network for phase-field modeling of fracture. Theoret. Appl. Fracture Mech. **106**, 102,447 (2020). https://doi.org/10.1016/j.tafmec.2019.102447, https://www.sciencedirect.com/science/article/pii/S016784421930357X
53. Grandits, T., Pezzuto, S., Costabal, F.S., et al.: Learning Atrial Fiber Orientations and Conductivity Tensors from Intracardiac Maps Using Physics-Informed Neural Networks. In: Ennis, D.B., Perotti, L.E., Wang, V.Y. (eds) Functional Imaging and Modeling of the Heart. Springer International Publishing, Cham, Lecture Notes in Comput. Sci., pp. 650–658 (2021), https://doi.org/10.1007/978-3-030-78710-3_62
54. Grubišić, L., Hajba, M., Lacmanović, D.: Deep Neural Network Model for Approximating Eigenmodes Localized by a Confining Potential. Entropy **23**(1), 95 (2021). https://doi.org/10.3390/e23010095, www.mdpi.com/1099-4300/23/1/95
55. Haghighat, E., Juanes, R.: SciANN: A Keras/Tensorflow wrapper for scientific computations and physics-informed deep learning using artificial neural networks. Comput. Methods Appl. Mech. Engrg. **373**, 113,552 (2021). https://doi.org/10.1016/j.cma.2020.113552, arXiv: 2005.08803

36. 德·里克，T.，米什拉，S.：物理信息神经网络 (PINNs) 逼近科尔莫哥洛夫偏微分方程的误差分析。(2021) arXiv:2106.14473 [cs, math]
37. 德·里克，T.，兰塔勒，S.，米什拉，S.：关于通过tanh神经网络逼近函数。神经网络 **143**， 732–750 (2021)。https://doi.org/10.1016/j.neunet.2021.08.015,www.sciencedirect.com/science/article/pii/S0893608021003208
38. 德·里克，T.，贾塔普，A.D.，米什拉，S.：逼近纳维-斯托克斯方程的物理信息神经网络的误差估计。(2022)arXiv:2203.09346 [cs, math]
39. 迪桑纳亚克 M.W.M.G.，范天 N.：基于神经网络的偏微分方程求解近似方法。工程数值方法杂志 **10**(3)，195–201 (1994). https://doi.org/10.1002/cnm.1640100303， https://onlinelibrary.wiley.com/doi/abs/10.1002/cnm.1640100303
40. 德里斯科尔 T.A.，黑尔 N.，特雷菲恩 L.N.：切比雪夫函数指南。Pafnuty 出版社， http://www.chebfun . org/docs/guide/ (2014)
41. Dwivedi V.，Srinivasan B.：物理信息极限学习机 (PIELM)-一种用于偏微分方程数值求解的快速方法。神经计算 **391**, 96–118 (2020). https://doi.org/10.1016/j.neucom.2019.12.099， www.sciencedirect.com/science/article/pii/S0925231219318144
42. EW, 余. B.: 深度里茨方法：一种基于深度学习的数值算法用于求解变分问题。数学统计通讯 **6**(1),1–12 (2018). https://doi.org/10.1007/s40304-018-0127-z
43. 埃尔布雷希特, D., 佩雷克雷斯特恩科, D., 格罗斯, P., 等：深度神经网络逼近理论。IEEE 信息论汇刊 **67**(5), 2581–2623(2021). https://doi.org/10.1109/TIT.2021.3062161
44. 方, Z.: 基于卷积神经网络的高效混合物理信息神经网络。神经网络与学习系统汇刊 pp. 1–13。(2021)https://doi.org/10.1109/TNNLS.2021.3070878
45. 方, Z., 战, J.: 用于超材料设计的深度物理信息神经网络。IEEE Access**8**,24506–24513(2020). https://doi.org/10.1109/ACCESS.2019.2963375
46. 方，Z., 战，J.: 物理信息神经网络框架用于三维表面上的偏微分方程：时间无关问题。IEEE Access **8**, 26328–26335(2020). https://doi.org/10.1109/ACCESS.2019.2963390
47. 富克斯，O., 切列皮，H.A.: 物理信息机器学习的局限性：多孔介质中的非线性两相传输。计算物理杂志 **1**(1) (2020). https://doi.org/10.1615/.2020033905,https://www.dl.begellhouse.com/journals/558048804a15188a,583c4e56625ba94e,415f83b5707fde65.html
48. 高，H., 孙，L., 王，J.X.: PhyGeoNet：物理信息几何自适应卷积神经网络，用于求解不规则区域上的参数化稳态偏微分方程。计算物理杂志**428**, 110,079(2021). https://doi.org/10.1016/j.jcp.2020.110079,https://www.sciencedirect.com/science/article/pii/S0021999120308536
49. 加德纳，J.R.，普莱斯，G.，宾德尔，D.，等：Gpytorch：具有GPU加速的黑盒矩阵-矩阵高斯过程推理。载于《神经信息处理系统进展》（2018）
50. 加尔内洛，M.，沙南汉，M.：协调深度学习与符号人工智能：表示对象和关系。行为科学当前观点**29**，17–23（2019）。https://doi.org/10.1016/j.cobeha.2018.12.010， www.sciencedirect.com/science/article/pii/S2352154618301943
51. 日内瓦，N.，扎巴拉斯，N.：使用物理约束深度自回归网络建模偏微分方程系统的动力学。计算物理杂志**403**, 109,056(2020)。https://doi.org/10.1016/j.jcp.2019.109056,https://www.sciencedirect.com/science/article/pii/S0021999119307612
52. 戈斯瓦米，S.， 阿尼泰斯科，C.，查克拉博尔蒂，S.，等：迁移学习增强物理信息神经网络用于断裂的相场建模。理论应用断裂力学**106**， 102,447 (2020).https://doi.org/10.1016/j.tafmec.2019.102447, https://www.sciencedirect.com/science/article/pii/S016784421930357X
53. 格兰迪茨，T.，佩佐托，S.，科斯塔巴尔，F.S.，等：使用物理信息神经网络从心内图中学习心房纤维方向和导率张量。在：恩尼斯，D.B.，佩罗蒂，L.E.，王，V.Y. (eds) 心脏的功能成像和建模。斯普林格国际出版，查姆，计算机科学讲义，第 650–658 页 (2021), https://doi.org/10.1007/978-3-030-78710-3_62
54. Grubiši´c, L., Hajba, M., Lacmanovi´c, D.: 深度神经网络模型用于逼近由约束势局部化的本征模式。熵 **23**(1), 95 (2021)。 https://doi.org/10.3390/e23010095,www.mdpi.com/1099-4300/23/1/95
55. Haghighat, E., Juanes, R.: SciANN: 一个用于科学计算和信息融合深度学习的 Keras/Tensorflow 人工神经网络包装器。计算方法与应用力学工程 **373**, 113,552 (2021)。 https://doi.org/10.1016/j.cma.2020.113552, arXiv: 2005.08803

56. Haghighat, E., Bekar, A.C., Madenci, E., et al.: A nonlocal physics-informed deep learning framework using the peridynamic differential operator. Comput. Methods Appl. Mech. Engrg. **385**, 114,012 (2021a). https://doi.org/10.1016/j.cma.2021.114012, https://www.sciencedirect.com/science/article/pii/S0045782521003431
57. Haghighat, E., Raissi, M., Moure, A., et al.: A physics-informed deep learning framework for inversion and surrogate modeling in solid mechanics. Comput. Methods Appl. Mech. Engrg. **379**, 113,741 (2021b). https://doi.org/10.1016/j.cma.2021.113741, https://www.sciencedirect.com/science/article/pii/S0045782521000773
58. Haitsiukevich, K., Ilin, A.: Improved Training of Physics-Informed Neural Networks with Model Ensembles. (2022) arXiv:2204.05108 [cs, stat]
59. He, Q., Tartakovsky, A.M.: Physics-informed neural network method for forward and backward advection-dispersion equations. Water Resources Research **57**(7), e2020WR029,479 (2021). https://doi.org/10.1029/2020WR029479, https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2020WR029479, e2020WR029479 2020WR029479
60. He, Q., Barajas-Solano, D., Tartakovsky, G., et al.: Physics-informed neural networks for multiphysics data assimilation with application to subsurface transport. Adv. Water Resources **141**, 103,610 (2020). https://doi.org/10.1016/j.advwatres.2020.103610, https://www.sciencedirect.com/science/article/pii/S0309170819311649
61. Hennigh, O., Narasimhan, S., Nabian, M.A., et al.: NVIDIA SimNet: An AI-Accelerated Multi-Physics Simulation Framework. In: Paszynski M, Kranzlmüller D, Krzhizhanovskaya VV, et al (eds) Computational Science – ICCS 2021. Springer International Publishing, Cham, Lecture Notes in Comput. Sci., pp. 447–461 (2021), https://doi.org/10.1007/978-3-030-77977-1_36
62. Hillebrecht, B., Unger, B.: Certified machine learning: A posteriori error estimation for physics-informed neural networks. Tech. rep., (2022) https://doi.org/10.48550/arXiv.2203.17055, arXiv:2203.17055 [cs, math] type: article
63. Hinze, M., Pinnau, R., Ulbrich, M., et al.: Optimization with PDE constraints, vol. 23. Springer Science & Business Media, Berlin (2008)
64. Hoffer, J.G., Geiger, B.C., Ofner, P., et al.: Mesh-Free Surrogate Models for Structural Mechanic FEM Simulation: A Comparative Study of Approaches. Appl. Sci. **11**(20), 9411 (2021). https://doi.org/10.3390/app11209411, www.mdpi.com/2076-3417/11/20/9411
65. Hornik, K., Stinchcombe, M., White, H.: Multilayer feedforward networks are universal approximators. Neural Networks **2**(5), 359–366 (1989). https://doi.org/10.1016/0893-6080(89)90020-8, www.sciencedirect.com/science/article/pii/0893608089900208
66. Huang, G.B., Wang, D.H., Lan, Y.: Extreme learning machines: a survey. Int. J. Mach. Learn. Cybern. **2**(2), 107–122 (2011). https://doi.org/10.1007/s13042-011-0019-y
67. Huang, X., Liu, H., Shi, B., et al.: Solving Partial Differential Equations with Point Source Based on Physics-Informed Neural Networks. (2021) arXiv:2111.01394 [physics]
68. Irrgang, C., Boers, N., Sonnewald, M., et al.: Towards neural Earth system modelling by integrating artificial intelligence in Earth system science. Nat. Mach. Intelligence **3**(8), 667–674 (2021). https://doi.org/10.1038/s42256-021-00374-3, www.nature.com/articles/s42256-021-00374-3
69. Islam, M., Thakur, M.S.H., Mojumder, S., et al.: Extraction of material properties through multi-fidelity deep learning from molecular dynamics simulation. Comput. Mater. Sci. **188**, 110,187 (2021). https://doi.org/10.1016/j.commatsci.2020.110187, https://www.sciencedirect.com/science/article/pii/S0927025620306789
70. Jagtap, A.D., Kawaguchi, K., Karniadakis, G.E.: Adaptive activation functions accelerate convergence in deep and physics-informed neural networks. J. Comput. Phys. **404**, 109,136 (2020a). https://doi.org/10.1016/j.jcp.2019.109136, https://www.sciencedirect.com/science/article/pii/S0021999119308411
71. Jagtap, A.D., Kharazmi, E., Karniadakis, G.E.: Conservative physics-informed neural networks on discrete domains for conservation laws: Applications to forward and inverse problems. Comput. Methods Appl. Mech. Engrg. **365**, 113,028 (2020b). https://doi.org/10.1016/j.cma.2020.113028, https://www.sciencedirect.com/science/article/pii/S0045782520302127
72. Jamali, B., Haghighat, E., Ignjatovic, A., et al.: Machine learning for accelerating 2D flood models: Potential and challenges. Hydrological Processes **35**(4), e14,064 (2021). https://doi.org/10.1002/hyp.14064, https://onlinelibrary.wiley.com/doi/abs/10.1002/hyp.14064
73. Jin, X., Cai, S., Li, H., et al.: NSFnets (Navier-Stokes flow nets): Physics-informed neural networks for the incompressible Navier-Stokes equations. J. Comput. Phys. **426**, 109,951 (2021). https://doi.org/10.1016/j.jcp.2020.109951, https://www.sciencedirect.com/science/article/pii/S0021999120307257
74. Karniadakis, G.E., Kevrekidis, I.G., Lu, L., et al.: Physics-informed machine learning. Nature Reviews Phys. **3**(6), 422–440 (2021). https://doi.org/10.1038/s42254-021-00314-5, www.nature.com/articles/s42254-021-00314-5

56. 哈吉加特，E.，贝卡，A.C.，马达尼，E.，等：基于本构微分算子的非局部物理信息深度学习框架。计算方法与应用力学工程 **385**，114,012(2021a)。 https://doi.org/10.1016/j.cma.2021.114012, https://www.sciencedirect.com/science/article/pii/S0045782521003431
57. 哈吉加特，E.，拉西，M.，莫尔，A.，等：用于固体力学的反演和代理建模的物理信息深度学习框架。计算方法与应用力学工程 **379**，113,741 (2021b)。 https://doi.org/10.1016/j.cma.2021.113741,https://www.sciencedirect.com/science/article/pii/S0045782521000773
58. 哈伊茨尤克维奇，K.，伊林，A.：基于模型集成改进物理信息神经网络的训练。(2022) arXiv:2204.05108 [cs, stat]
59. 何，Q.，塔塔科夫斯基，A.M.：用于正向和反向平流弥散方程的物理信息神经网络方法。水资源研究 **57**(7)，e2020WR029，479 (2021)。 https://doi.org/10.1029/2020WR029479, https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2020WR029479，e2020WR029479 2020WR029479
60. 何，Q.，巴拉哈斯-索洛诺，D.，塔塔科夫斯基，G.，等：用于多物理场数据同化的物理信息神经网络，应用于地下传输。先进水资源杂志**141**，103，610 (2020)。 https://doi.org/10.1016/j.advwatres.2020.103610, https://www.sciencedirect.com/science/article/pii/S0309170819311649
61. 亨尼希，O.，纳拉辛汉，S.，纳比安，M.A.，等：NVIDIA SimNet：AI加速多物理场仿真框架。载于：帕辛斯基，M.，克兰兹穆勒，D.，克日希扎诺夫斯基娅，VV.，等（编）计算科学 – ICCS 2021。斯普林格国际出版公司，查姆，计算机科学讲义，第447–461页（2021），https://doi.org/10.1007/978-3-030-77977-1_36
62. 希勒布雷希特，B.，乌nger，B.：认证机器学习：物理信息神经网络的后验误差估计。技术报告，（2022）https://doi.org/10.48550/arXiv.2203.17055,arXiv:2203.17055 [cs,math]类型：文章
63. Hinze, M.，Pinnau, R.，Ulbrich, M.，等：使用PDE约束的优化，第23卷。SpringerScience & Business Media，柏林（2008）
64. Hoffer, J.G., Geiger, B.C., Ofner, P., 等：无网格代理模型用于结构力学FEM模拟：方法的比较研究。应用科学 **11**(20), 9411 (2021).https://doi.org/10.3390/app11209411, www.mdpi.com/2076-3417/11/20/9411
65. Hornik, K., Stinchcombe, M., White, H.: 多层前馈网络是通用逼近器。神经网络 **2**(5)，359–366 (1989). https://doi.org/10.1016/0893-6080(89)90020-8,www.sciencedirect.com/science/article/pii/0893608089900208
66. Huang, G.B., Wang, D.H., Lan, Y.: 极限学习机：综述。国际机器学习与计算智能杂志**2**(2), 107–122(2011). https://doi.org/10.1007/s13042-011-0019-y
67. Huang, X., Liu, H., Shi, B., 等：基于物理信息神经网络的点源求解偏微分方程。(2021) arXiv:2111.01394 [物理]
68. Irrgang, C., Boers, N., Sonnewald, M., 等：通过在地球系统科学中集成人工智能实现神经地球系统建模。自然机器智能 **3**(8), 667–674(2021)。 https://doi.org/10.1038/s42256-021-00374-3, www.nature.com/articles/s42256-021-00374-3
69. Islam, M., Thakur, M.S.H., Mojumder, S., 等：通过分子动力学模拟的多保真度深度学习提取材料属性。计算材料科学 **188**, 110,187(2021).https://doi.org/10.1016/j.commatsci.2020.110187, https://www.sciencedirect.com/science/article/pii/S0927025620306789
70. 贾塔普，A.D.，川口，K.，卡尼亚达基斯，G.E.：自适应激活函数加速深度和物理信息神经网络中的收敛性。计算物理杂志**404**, 109，136(2020a)。 https://doi.org/10.1016/j.jcp.2019.109136， https://www.sciencedirect.com/science/article/pii/S0021999119308411
71. 贾塔普，A.D.，哈拉兹米，E.，卡尼亚达基斯，G.E.：离散域上的守恒物理信息神经网络，用于守恒律：正向和逆问题的应用。计算方法与应用力学工程**365**，113，028 (2020b)。 https://doi.org/10.1016/j.cma.2020.113028,https://www.sciencedirect.com/science/article/pii/S0045782520302127
72. 贾马里，B.，哈吉加特，E.，伊格尼亚托维奇，A.，等：用于加速二维洪水模型的机器学习：潜力和挑战。水文过程 **35**(4)，e14，064 (2021)。https://doi.org/10.1002/hyp.14064， https://onlinelibrary.wiley.com/doi/abs/10.1002/hyp.14064
73. 金，X.，蔡，S.，李，H.，等：NSFnets（纳维-斯托克斯流网）：物理信息神经网络用于不可压缩纳维-斯托克斯方程。计算物理杂志**426**，109,951 (2021)。https://doi.org/10.1016/j.jcp.2020.109951， https://www.sciencedirect.com/science/article/pii/S0021999120307257
74. Karniadakis, G.E.，凯夫雷基迪斯，I.G.，陆，L.，等：物理信息机器学习。自然物理评论**3**(6)，422–440 (2021)。https://doi.org/10.1038/s42254-021-00314-5,www.nature.com/articles/s42254-021-00314-5

75. Kashinath, K., Mustafa, M., Albert, A., et al.: Physics-informed machine learning: case studies for weather and climate modelling. Philosophical Transactions of the Royal Society A: Mathematical, Phys. Eng. Sci. **379**(2194), 20200,093 (2021). https://doi.org/10.1098/rsta.2020.0093, https://royalsocietypublishing.org/doi/full/10.1098/rsta.2020.0093
76. Kharazmi, E., Zhang, Z., Karniadakis, G.E.: Variational Physics-Informed Neural Networks For Solving Partial Differential Equations. (2019) arXiv:1912.00873 [physics, stat]
77. Kharazmi, E., Cai, M., Zheng, X., et al.: Identifiability and predictability of integer- and fractional-order epidemiological models using physics-informed neural networks. Nature Comput. Sci. **1**(11), 744–753 (2021). https://doi.org/10.1038/s43588-021-00158-0
78. Kharazmi, E., Zhang, Z., Karniadakis, G.E.M.: hp-VPINNs: Variational physics-informed neural networks with domain decomposition. Comput. Methods Appl. Mech. Engrg. **374**, 113,547 (2021b). https://doi.org/10.1016/j.cma.2020.113547, https://www.sciencedirect.com/science/article/pii/S0045782520307325
79. Kim, J., Lee, K., Lee, D., et al.: DPM: A Novel Training Method for Physics-Informed Neural Networks in Extrapolation. Proc. AAAI Conf. Artif. Intell. **35**(9), 8146–8154 (2021a). https://ojs.aaai.org/index.php/AAAI/article/view/16992
80. Kim, S.W., Kim, I., Lee, J., et al.: Knowledge Integration into deep learning in dynamical systems: an overview and taxonomy. J. Mech. Sci. Technol. **35**(4), 1331–1342 (2021). https://doi.org/10.1007/s12206-021-0342-5
81. Kissas, G., Yang, Y., Hwuang, E., et al.: Machine learning in cardiovascular flows modeling: Predicting arterial blood pressure from non-invasive 4D flow MRI data using physics-informed neural networks. Comput. Methods Appl. Mech. Engrg. **358**, 112,623 (2020). https://doi.org/10.1016/j.cma.2019.112623, https://www.sciencedirect.com/science/article/pii/S0045782519305055
82. Kollmannsberger, S., D'Angella, D., Jokeit, M., et al.: Physics-Informed Neural Networks. In: Kollmannsberger, S., D'Angella, D., Jokeit, M., et al. (eds.) Deep Learning in Computational Mechanics, pp. 55–84. Studies in Computational Intelligence, Springer International Publishing, Cham (2021). https://doi.org/10.1007/978-3-030-76587-3_5
83. Kondor, R., Trivedi, S.: On the Generalization of Equivariance and Convolution in Neural Networks to the Action of Compact Groups. In: Dy J, Krause A (eds) Proceedings of the 35th International Conference on Machine Learning, Proc. Mach. Learn. Res., vol **80**. PMLR, pp. 2747–2755 (2018), https://proceedings.mlr.press/v80/kondor18a.html
84. Koryagin, A., Khudorozkov, R., Tsimfer, S.: PyDEns: a Python Framework for Solving Differential Equations with Neural Networks. (2019) arXiv:1909.11544 [cs, stat]
85. Kovacs, A., Exl, L., Kornell, A., et al.: Conditional physics informed neural networks. Commun. Nonlinear Sci. Numer. Simulation **104**, 106,041 (2022). https://doi.org/10.1016/j.cnsns.2021.106041, https://www.sciencedirect.com/science/article/pii/S1007570421003531
86. Krishnapriyan, A., Gholami, A., Zhe, S., et al.: Characterizing possible failure modes in physics-informed neural networks. In: Ranzato, M., Beygelzimer, A., Dauphin, Y., et al (eds) Advances in Neural Information Processing Systems, vol **34**. Curran Associates, Inc., pp. 26,548–26,560 (2021), https://proceedings.neurips.cc/paper/2021/file/df438e5206f31600e6ae4af72f2725f1-Paper.pdf
87. Kumar, M., Yadav, N.: Multilayer perceptrons and radial basis function neural network methods for the solution of differential equations: A survey. Computers & Mathematics with Applications **62**(10), 3796–3811 (2011). https://doi.org/10.1016/j.camwa.2011.09.028, www.sciencedirect.com/science/article/pii/S0898122111007966
88. Kutyniok, G.: The Mathematics of Artificial Intelligence (2022). arXiv:2203.08890 [cs, math, stat]
89. Lagaris, I., Likas, A., Fotiadis, D.: Artificial neural networks for solving ordinary and partial differential equations. IEEE Trans. Neural Networks **9**(5), 987–1000 (1998). https://doi.org/10.1109/72.712178
90. Lagaris, I., Likas, A., Papageorgiou, D.: Neural-network methods for boundary value problems with irregular boundaries. IEEE Trans. Neural Networks **11**(5), 1041–1049 (2000). https://doi.org/10.1109/72.870037
91. Lai, Z., Mylonas, C., Nagarajaiah, S., et al.: Structural identification with physics-informed neural ordinary differential equations. J. Sound and Vibration **508**, 116,196 (2021). https://doi.org/10.1016/j.jsv.2021.116196, https://www.sciencedirect.com/science/article/pii/S0022460X21002686
92. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**(7553), 436–444 (2015). https://doi.org/10.1038/nature14539
93. Lee, H., Kang, I.S.: Neural algorithm for solving differential equations. J. Comput. Phys. **91**(1), 110–131 (1990). https://doi.org/10.1016/0021-9991(90)90007-N, www.sciencedirect.com/science/article/pii/002199919090007N
94. Li, W., Bazant, M.Z., Zhu, J.: A physics-guided neural network framework for elastic plates: Comparison of governing equations-based and energy-based approaches. Comput. Methods Appl. Mech.

75. Kashinath, K., Mustafa, M., Albert, A., 等：物理信息机器学习：天气和气候建模案例研究。皇家学会哲学汇刊A：数学、物理与工程科学 **379**(2194), 20200,093 (2021)。 https://doi.org/10.1098/rsta.2020.0093, https://royalsocietypublishing.org/doi/full/10.1098/rsta.2020.0093
76. Kharazmi, E., Zhang, Z., Karniadakis, G.E.：变分物理信息神经网络用于求解偏微分方程。（2019） arXiv:1912.00873 [物理，统计]
77. Kharazmi, E., Cai, M., Zheng, X., 等：使用物理信息神经网络识别和预测整数和分数阶流行病学模型。自然计算科学 **1**(11), 744–753 (2021)。 https://doi.org/10.1038/s43588-021-00158-0
78. 哈拉兹米, E., 张, Z., Karniadakis, G.E.M.: hp-VPINNs: 域分解的变分物理信息神经网络. 计算方法与应用力学工程 **374**，113,547(2021b). https://doi.org/10.1016/j.cma.2020.113547,https://www.sciencedirect.com/science/article/pii/S0045782520307325
79. Kim, J., 李, K., 李, D., 等.: DPM: 外推中物理信息神经网络的创新训练方法. AAAI人工智能会议论文集 **35**(9), 8146–8154(2021a). https://ojs.aaai.org/index.php/AAAI/article/view/16992
80. Kim, S.W., Kim, I., 李, J., 等.: 动力系统中的深度学习知识集成：概述和分类. 机械科学与技术杂志 **35**(4), 1331–1342(2021). https://doi.org/10.1007/s12206-021-0342-5
81. Kissas, G., Yang, Y., Hwuang, E., 等：机器学习在心血管流动建模中的应用：利用物理信息神经网络从非侵入性4D血流MRI数据预测动脉血压。计算方法与应用力学工程**358**，112,623 (2020)。 https://doi.org/10.1016/j.cma.2019.112623,https://www.sciencedirect.com/science/article/pii/S0045782519305055
82. Kollmannsberger, S., D'Angella, D., Jokeit, M., 等：物理信息神经网络。载于：Kollmannsberger,S., D'Angella, D., Jokeit, M., 等(eds.) 计算力学中的深度学习，第55–84页。计算智能研究，斯普林格国际出版公司，查姆 (2021)。 https://doi.org/10.1007/978-3-030-76587-3_5
83. 孔多尔，R., 特里维迪，S.: 关于等变性和卷积在神经网络中推广到紧致群的作用。在：迪，J，克劳斯，A（编）第35届国际机器学习会议论文集，机器学习研究进展，卷 **80**。 PMLR，第2747–2755页（2018年）， https://proceedings.mlr.press/v80/kondor18a.html
84. 科内尔，A., 胡多罗佐夫，R., 茨米尔费尔，S.: pydens：一个用于使用神经网络求解微分方程的Python框架。（2019年） arXiv:1909.11544 [cs, stat]
85. 科瓦奇，A., 埃克斯尔，L., 科内尔，A., 等：条件物理信息神经网络。非线性科学数值模拟通讯 **104**，第106,041页（2022年）。 https://doi.org/10.1016/j.cnsns.2021.106041,https://www.sciencedirect.com/science/article/pii/S1007570421003531
86. 克里希纳皮扬，A.，戈拉米，A.，泽，S.，等：表征物理信息神经网络中可能的失效模式。在：兰扎托，M.，贝格尔齐默，A.，杜菲，Y.，等 (eds) 神经信息处理系统进展，卷 **34**。库兰协会，pp. 26,548–26,560 (2021), https://proceedings. neurips.cc/paper/2021/file/df438e5206f31600e6ae4af72f2725f1-Paper.pdf
87. 库马尔, M.，亚达夫，N.：多层感知器和径向基函数神经网络方法用于微分方程的求解：综述。计算机与数学应用 **62**(10),3796– 3811 (2011)。 https://doi.org/10.1016/j.camwa.2011.09.028,www.sciencedirect.com/science/article/pii/S0898122111007966库蒂尼奥克，G.：人工智能数学 (2022)。 arXiv:2203.08890 [cs, math, stat]
88.
89. 拉加里斯，I.，利卡斯，A.，福蒂亚迪斯，D.：人工神经网络用于求解常微分方程和偏微分方程。IEEE神经网络汇刊**9**(5)，987–1000 (1998)。https://doi.org/10.1109/72.712178
90. 拉加里斯，I.，利卡斯，A.，帕帕格奥尔吉乌，D.：用于具有不规则边界的边值问题的神经网络方法。IEEE神经网络汇刊**11**(5)，1041–1049(2000)。https://doi.org/10.1109/72.870037
91. 莱，Z.，米隆纳斯，C.，纳加拉亚哈，S.，等：使用物理信息融合常微分方程的神经网络进行结构识别。声音与振动杂志**508**，116,196 (2021)。https://doi.org/10.1016/j.jsv.2021.116196， https://www.sciencedirect.com/science/article/pii/S0022460X21002686
92. 勒库恩, Y., 贝吉奥, Y., 希顿, G.: 深度学习. 自然 **521**(7553), 436–444(2015). https://doi.org/10.1038/nature14539
93. 李, H., 康, I.S.: 求解微分方程的神经算法. 计算物理杂志 **91**(1),110–131 (1990). https://doi.org/10.1016/0021-9991(90)90007-N,www.sciencedirect.com/science/article/pii/002199919090007N
94. 李, W., 巴赞特, M.Z., 朱, J.: 弹性板的物理引导神经网络框架：基于控制方程和基于能量的方法的比较. 计算方法与应用力学，

Engrg. **383**, 113,933 (2021). https://doi.org/10.1016/j.cma.2021.113933, https://www.sciencedirect.com/science/article/pii/S004578252100270X
95. Lin, C., Li, Z., Lu, L., et al.: Operator learning for predicting multiscale bubble growth dynamics. J. Chem. Phys. **154**(10), 104,118 (2021a). https://doi.org/10.1063/5.0041203, https://aip.scitation.org/doi/10.1063/5.0041203
96. Lin, C., Maxey, M., Li, Z., et al.: A seamless multiscale operator neural network for inferring bubble dynamics. J. Fluid Mech. 929 (2021b). https://doi.org/10.1017/jfm.2021.866, https://www.cambridge.org/core/journals/journal-of-fluid-mechanics/article/seamless-multiscale-operator-neural-network-for-inferring-bubble-dynamics/D516AB0EF954D0FF56AD864DB2618E94
97. Liu, D., Wang, Y.: A Dual-Dimer method for training physics-constrained neural networks with minimax architecture. Neural Netw. **136**, 112–125 (2021). https://doi.org/10.1016/j.neunet.2020.12.028, www.sciencedirect.com/science/article/pii/S0893608020304536
98. Lu, L., Dao, M., Kumar, P., et al.: Extraction of mechanical properties of materials through deep learning from instrumented indentation. Proc. Nat. Acad. Sci. India Sect. **117**(13), 7052–7062 (2020). https://doi.org/10.1073/pnas.1922210117, www.pnas.org/content/117/13/7052
99. Lu, L., Jin, P., Pang, G., et al.: Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. Nat. Mac. Intell. **3**(3), 218–229 (2021). https://doi.org/10.1038/s42256-021-00302-5, www.nature.com/articles/s42256-021-00302-5
100. Lu, L., Meng, X., Mao, Z., et al.: DeepXDE: A deep learning library for solving differential equations. SIAM Rev. **63**(1), 208–228 (2021). https://doi.org/10.1137/19M1274067
101. Lu, L., Pestourie, R., Yao, W., et al.: Physics-informed neural networks with hard constraints for inverse design. SIAM J. Sci. Comput. **43**(6), B1105–B1132 (2021). https://doi.org/10.1137/21M1397908
102. Mallat, S.: Understanding deep convolutional networks. Philosophical Transactions of the Royal Society A: Mathematical, Phys. Eng. Sci. **374**(2065), 20150,203 (2016). https://doi.org/10.1098/rsta.2015.0203, https://royalsocietypublishing.org/doi/10.1098/rsta.2015.0203
103. Mao, Z., Jagtap, A.D., Karniadakis, G.E.: Physics-informed neural networks for high-speed flows. Comput. Methods Appl. Mech. Engrg. **360**, 112,789 (2020). https://doi.org/10.1016/j.cma.2019.112789, https://www.sciencedirect.com/science/article/pii/S0045782519306814
104. Mao, Z., Lu, L., Marxen, O., et al.: DeepM&Mnet for hypersonics: Predicting the coupled flow and finite-rate chemistry behind a normal shock using neural-network approximation of operators. J. Comput. Phys. **447**, 110,698 (2021). https://doi.org/10.1016/j.jcp.2021.110698, https://www.sciencedirect.com/science/article/pii/S0021999121005933
105. Markidis, S.: The Old and the New: Can Physics-Informed Deep-Learning Replace Traditional Linear Solvers? Frontiers in Big Data 4 (2021). https://www.frontiersin.org/article/10.3389/fdata.2021.669097
106. Mathews, A., Francisquez, M., Hughes, J.W., et al.: Uncovering turbulent plasma dynamics via deep learning from partial observations. Phys. Review E **104**(2) (2021). https://doi.org/10.1103/physreve.104.025205, https://www.osti.gov/pages/biblio/1813020
107. McClenny, L.D., Haile, M.A., Braga-Neto, U.M.: Tensordiffeq: Scalable multi-gpu forward and inverse solvers for physics informed neural networks. (2021) arXiv preprint arXiv:2103.16034
108. Mehta, P.P., Pang, G., Song, F., et al.: Discovering a universal variable-order fractional model for turbulent couette flow using a physics-informed neural network. Fract. Calc. Appl. Anal. **22**(6), 1675–1688 (2019). https://doi.org/10.1515/fca-2019-0086
109. Meng, X., Li, Z., Zhang, D., et al.: Ppinn: Parareal physics-informed neural network for time-dependent pdes. Comput. Methods Appl. Mech. Engrg. **370**, 113,250 (2020). https://doi.org/10.1016/j.cma.2020.113250, https://www.sciencedirect.com/science/article/pii/S0045782520304357
110. Minh Nguyen-Thanh, V., Trong Khiem Nguyen, L., Rabczuk, T., et al.: A surrogate model for computational homogenization of elastostatics at finite strain using high-dimensional model representation-based neural network. Int. J. Numer. Methods Eng. **121**(21), 4811–4842 (2020). https://doi.org/10.1002/nme.6493, https://onlinelibrary.wiley.com/doi/abs/10.1002/nme.6493
111. Mishra, S., Molinaro, R.: Estimates on the generalization error of physics-informed neural networks for approximating a class of inverse problems for PDEs. IMA J. Numer. Anal. (2021). https://doi.org/10.1093/imanum/drab032
112. Mishra, S., Molinaro, R.: Physics informed neural networks for simulating radiative transfer. J. Quant. Spectroscopy and Radiative Transf. **270**, 107,705 (2021b). https://doi.org/10.1016/j.jqsrt.2021.107705, https://www.sciencedirect.com/science/article/pii/S0022407321001989
113. Mishra, S., Molinaro, R.: Estimates on the generalization error of physics-informed neural networks for approximating PDEs. IMA J. Numer. Anal. p drab093 (2022). https://doi.org/10.1093/imanum/drab093
114. Misyris, G.S., Venzke, A., Chatzivasileiadis, S.: Physics-informed neural networks for power systems. 2020 IEEE Power & Energy Society General Meeting (PESGM) pp. 1–5 (2020)

Engrg. **383**, 113,933 (2021). https://doi.org/10.1016/j.cma.2021.113933, https://www.sciencedirect.com/science/article/pii/S004578252100270X
95. 林，C.，李，Z.，陆，L.，等：算子学习用于预测多尺度气泡生长动力学。化学物理杂志**154**(10), 104,118 (2021a). https://doi.org/10.1063/5.0041203, https://aip.scitation.org/doi/10.1063/5.0041203
96. 林，C.，马克西，M.，李，Z.，等：无缝多尺度算子神经网络用于推断气泡动力学。流体力学杂志 929 (2021b). https://doi.org/10.1017/jfm.2021.866,https://www.cambridge.org/core/journals/journal-of-fluid-mechanics/article/seamless-multiscale-operator-neural-network-for-inferring-bubble-dynamics/D516AB0EF954D0FF56AD864DB2618E94
97. 刘，D.，王，Y.：一种基于极小极大架构的双二聚体方法，用于训练物理约束神经网络。神经网络 **136**， 112–125 (2021)。 https://doi.org/10.1016/j.neunet.2020.12.028， www.sciencedirect.com/science/article/pii/S0893608020304536
98. 陆，L.，道，M.，Kumar,P.，等：通过深度学习从仪器压痕中提取材料力学性能。印度国家科学院公报 **117**(13)， 7052–7062 (2020)。 https://doi.org/10.1073/pnas.1922210117， www.pnas.org/content/117/13/7052
99. 陆，L.，金，P.，庞，G.，等：基于算子通用逼近定理的DeepONet学习非线性算子。自然·机器智能 **3**(3), 218–229(2021)。 https://doi.org/10.1038/s42256-021-00302-5， www.nature.com/articles/s42256-021-00302-5
100. 陆，L.，孟，X.，毛，Z.，等：DeepXDE：一个用于求解微分方程的深度学习库。SIAM Rev. **63**(1), 208–228 (2021).https://doi.org/10.1137/19M1274067陆，L.，佩斯托里，R.，姚，W.，等：具有硬约束的物理信息神经网络用于逆
101.
design. SIAM J. Sci. Comput. **43**(6), B1105–B1132 (2021). https://doi.org/10.1137/21M1397908
102. 马拉丁，S.：深度卷积网络的理解。皇家学会哲学汇刊A：数学、物理与工程科学 **374**(2065), 20150,203 (2016).https://doi.org/10.1098/rsta.2015.0203,https://royalsocietypublishing.org/doi/10.1098/rsta.2015.0203
103. 毛，Z.，贾塔普，A.D.，Karniadakis, G.E.：用于高速流的物理信息神经网络。计算方法与应用力学工程**360**, 112,789 (2020). https://doi.org/10.1016/j.cma.2019.112789,https://www.sciencedirect.com/science/article/pii/S0045782519306814
104. 毛，Z.，陆，L.，马克思恩，O.，等：DeepM&Mnet用于高超音速：使用算子神经网络近似预测正激波后的耦合流和有限速率化学反应。计算物理学报 **447**， 110,698 (2021)。 https://doi.org/10.1016/j.jcp.2021.110698,https://www.sciencedirect.com/science/article/pii/S0021999121005933
105. 马克迪斯，S.：旧与新的：物理信息深度学习能否取代传统线性求解器？大数据前沿 4 (2021)。 https://www.frontiersin.org/article/10.3389/fdata.2021.669097
106. 马修斯，A.，弗朗西斯奎兹，M.，休斯，J.W.，等：通过深度学习从部分观测中揭示湍流等离子体动力学。物理评论E **104**(2) (2021)。https://doi.org/10.1103/physreve.104.025205， https://www.osti.gov/pages/biblio/1813020
107. 麦克伦尼，L.D.，海勒，M.A.，Braga-Neto,U.M.：Tensordiffeq：可扩展的多GPU正向和逆求解器，用于物理信息神经网络。(2021) arXiv preprint arXiv:2103.16034
108. 梅塔，P.P.，庞，G.，宋，F.，等：使用物理信息神经网络发现湍流库埃特流的通用变量阶分数模型。分数微积分与应用分析 **22**(6)，1675–1688 (2019)。https://doi.org/10.1515/fca-2019-0086
109. 孟，X.，李，Z.，张,D.，等：Ppinn：用于时变偏微分方程的并行物理信息神经网络。计算方法与应用力学工程 **370**， 113,250 (2020)。 https://doi.org/10.1016/j.cma.2020.113250， https://www.sciencedirect.com/science/article/pii/S0045782520304357
110. 明·阮氏·成，V.，阮氏·清·琴，L.，拉夫丘克，T.，等：一种基于高维模型表示的神经网络代理模型，用于计算有限应变下的弹性静力学均匀化。国际数值方法工程杂志**121**(21)，4811–4842 (2020)。https://doi.org/10.1002/nme.6493， https://onlinelibrary.wiley.com/doi/abs/10.1002/nme.6493
111. 米什拉，S.，莫利纳罗，R.：物理信息神经网络逼近偏微分方程逆问题类的一般化误差估计。国际数学分析杂志 (2021)。https://doi.org/10.1093/imanum/drab032
112. 米什拉，S.，莫利纳罗，R.：物理信息神经网络用于模拟辐射传输。量子光谱学与辐射传输杂志**270**，107,705 (2021b)。 https://doi.org/10.1016/j.jqsrt.2021.107705,https://www.sciencedirect.com/science/article/pii/S0022407321001989
113. 米什拉·S，莫利纳罗·R：物理信息神经网络逼近偏微分方程的泛化误差估计。国际数学分析杂志 p drab093 (2022年)。 https://doi.org/10.1093/imanum/drab093
114. 米西里斯·G·S，文茨克·A，查齐瓦西莱亚迪斯·S：电力系统的物理信息神经网络。2020IEEE电力与能源学会通用会议（PESGM） pp. 1–5 (2020)

115. Mo, Y., Ling, L., Zeng, D.: Data-driven vector soliton solutions of coupled nonlinear Schrödinger equation using a deep learning algorithm. Phys. Lett. A **421**, 127,739 (2022). https://doi.org/10.1016/j.physleta.2021.127739, https://www.sciencedirect.com/science/article/pii/S0375960121006034
116. Moseley, B., Markham, A., Nissen-Meyer, T.: Finite Basis Physics-Informed Neural Networks (FBPINNs): a scalable domain decomposition approach for solving differential equations. (2021) arXiv:2107.07871 [physics]
117. Muhammad, A.N., Aseere, A.M., Chiroma, H., et al.: Deep learning application in smart cities: recent development, taxonomy, challenges and research prospects. Neural Comput. Appl. **33**(7), 2973–3009 (2021). https://doi.org/10.1007/s00521-020-05151-8
118. Nabian, M.A., Gladstone, R.J., Meidani, H.: Efficient training of physics-informed neural networks via importance sampling. Comput. Aided Civil Infrastruct. Eng. **36**(8), 962–977 (2021). https://doi.org/10.1111/mice.12685, https://onlinelibrary.wiley.com/doi/abs/10.1111/mice.12685
119. Nandi, T., Hennigh, O., Nabian, M., et al.: Progress Towards Solving High Reynolds Number Reacting Flows in SimNet. Tech. rep., (2021) https://www.osti.gov/biblio/1846970-progress-towards-solving-high-reynolds-number-reacting-flows-simnet
120. Nandi, T., Hennigh, O., Nabian, M., et al.: Developing Digital Twins for Energy Applications Using Modulus. Tech. rep., (2022) https://www.osti.gov/biblio/1866819
121. Nascimento, R.G., Fricke, K., Viana, F.A.: A tutorial on solving ordinary differential equations using python and hybrid physics-informed neural network. Eng. Appl. Artif. Intell. **96**, 103,996. (2020) https://doi.org/10.1016/j.engappai.2020.103996, https://www.sciencedirect.com/science/article/pii/S095219762030292X
122. Novak, R., Xiao, L., Hron, J., et al.: Neural tangents: Fast and easy infinite neural networks in python. In: International Conference on Learning Representations, (2020) https://github.com/google/neural-tangents
123. NVIDIA Corporation (2021) Modulus User Guide. https://developer.nvidia.com/modulus-user-guide-v2106, release v21.06 – November 9, (2021)
124. Oreshkin, B.N., Carpov, D., Chapados, N., et al.: N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. (2020) arXiv:1905.10437 [cs, stat]
125. Owhadi, H.: Bayesian numerical homogenization. Multiscale Model. Simul **13**(3), 812–828 (2015). https://doi.org/10.1137/140974596
126. Owhadi, H., Yoo, G.R.: Kernel Flows: From learning kernels from data into the abyss. J. Comput. Phys. **389**, 22–47 (2019). https://doi.org/10.1016/j.jcp.2019.03.040, www.sciencedirect.com/science/article/pii/S0021999119302232
127. Özbay, A.G., Hamzehloo, A., Laizet, S., et al.: Poisson CNN: Convolutional neural networks for the solution of the Poisson equation on a Cartesian mesh. Data-Centric Engineering 2. (2021) https://doi.org/10.1017/dce.2021.7, https://www.cambridge.org/core/journals/data-centric-engineering/article/poisson-cnn-convolutional-neural-networks-for-the-solution-of-the-poisson-equation-on-a-cartesian-mesh/8CDFD5C9D5172E51B924E9AA1BA253A1
128. Pang, G., Lu, L., Karniadakis, G.E.: fPINNs: Fractional Physics-Informed Neural Networks. SIAM J. Sci. Comput. **41**(4), A2603–A2626 (2019). https://doi.org/10.1137/18M1229845, https://epubs.siam.org/doi/abs/10.1137/18M1229845
129. Paszke, A., Gross, S., Chintala, S., et al.: Automatic differentiation in PyTorch. Tech. rep., (2017) https://openreview.net/forum?id=BJJsrmfCZ
130. Patel, R.G., Manickam, I., Trask, N.A., et al.: Thermodynamically consistent physics-informed neural networks for hyperbolic systems. J. Comput. Phys. **449**, 110,754 (2022). https://doi.org/10.1016/j.jcp.2021.110754, https://www.sciencedirect.com/science/article/pii/S0021999121006495
131. Pedro, J.B., Maroñas, J., Paredes, R.: Solving Partial Differential Equations with Neural Networks. (2019) arXiv:1912.04737 [physics]
132. Peng, W., Zhang, J., Zhou, W., et al.: IDRLnet: A Physics-Informed Neural Network Library. (2021) arXiv:2107.04320 [cs, math]
133. Pinkus, A.: Approximation theory of the MLP model in neural networks. Acta Numer. **8**, 143–195 (1999). https://doi.org/10.1017/S0962492900002919, www.cambridge.org/core/journals/acta-numerica/article/abs/approximation-theory-of-the-mlp-model-in-neural-networks/18072C558C8410C4F92A82BCC8FC8CF9
134. Pratama, D.A., Bakar, M.A., Man, M., et al.: ANNs-Based Method for Solving Partial Differential Equations : A Survey. (2021) Preprint https://doi.org/10.20944/preprints202102.0160.v1, https://www.preprints.org/manuscript/202102.0160/v1
135. Psichogios, D.C., Ungar, L.H.: A hybrid neural network-first principles approach to process modeling. AIChE J. **38**(10), 1499–1511 (1992). https://doi.org/10.1002/aic.690381003, https://onlinelibrary.wiley.com/doi/abs/10.1002/aic.690381003

115. Mo, Y., Ling, L., Zeng, D.: 数据驱动深度学习算法求解耦合非线性薛定谔方程的矢量孤子解。物理快报A **421**，127,739 (2022)。 https://doi.org/10.1016/j.physleta.2021.127739， https://www.sciencedirect.com/science/article/pii/S0375960121006034
116. Moseley, B., Markham, A., Nissen-Meyer, T.: 有限基物理信息神经网络 (FBPINNs)：一种可扩展的域分解方法用于求解微分方程。(2021)arXiv:2107.07871 [物理]
117. Muhammad, A.N., Aseere, A.M., Chiroma, H., 等：深度学习在城市智能中的应用：最新发展、分类、挑战和研究前景。神经计算应用 **33**(7)，2973–3009 (2021)。 https://doi.org/10.1007/s00521-020-05151-8
118. Nabian, M.A., Gladstone, R.J., Meidani, H.: Efficient training of physics-informed neural networks via importance sampling. 计算机辅助土木工程 **36**(8), 962–977 (2021).https://doi.org/10.1111/mice.12685, https://onlinelibrary.wiley.com/doi/abs/10.1111/mice.12685
119. Nandi, T., 亨尼希，O., Nabian, M., 等：Progress Towards Solving High Reynolds Number Reacting Flows in SimNet. 技术报告，(2021)https://www.osti.gov/biblio/1846970-progress-towards-solving-high-reynolds-number-reacting-flows-simnet
120. Nandi, T., 亨尼希，O., Nabian, M., 等：Developing Digital Twins for Energy Applications Using Modulus. 技术报告，(2022)https://www.osti.gov/biblio/1866819
121. 纳西メント，R.G.，弗里克，K.，维安纳，F.A.：使用python和混合物理信息神经网络的常微分方程教程。工程应用人工智能**96**，103,996。(2020) https://doi.org/10.1016/j.engappai.2020.103996,https://www.sciencedirect.com/science/article/pii/S095219762030292X
122. 诺瓦克，R.，肖，L.，霍恩，J.，等：神经切线：python中快速且简单的无限神经网络 . 在：学习表示国际会议，(2020) https://github.com/google/neural-tangents
123. 英伟达公司 (2021) Modulus用户指南。 https://developer.nvidia.com/modulus-user-guide-v2106，发布版本v21.06 – 2021年11月9日
124. 奥雷什金，B.N.，卡普夫，D.，查帕多斯，N.，等：N-BEATS：用于可解释时间序列预测的神经基础扩展分析。(2020)arXiv:1905.10437 [cs, stat]
125. 奥瓦迪，H.：贝叶斯数值均匀化。多尺度模型模拟 **13**(3),812–828 (2015).https://doi.org/10.1137/140974596
126. 奥瓦迪，H.，尤，G.R.：核流：从数据中学习核进入深渊。计算物理杂志**389**, 22–47(2019). https://doi.org/10.1016/j.jcp.2019.03.040, www.sciencedirect.com/science/article/pii/S0021999119302232
127. 奥兹贝伊，A.G.，哈姆泽洛，A.，拉泽特，S.，等：泊松CNN：卷积神经网络用于笛卡尔网格上的泊松方程求解。数据中心工程 2. (2021)https://doi.org/10.1017/dce.2021.7, https://www.cambridge.org/core/journals/data-centric-engineering/article/poisson-cnn-convolutional-neural-networks-for-the-solution-of-the-poisson-equation-on-a-cartesian-mesh/8CDFD5C9D5172E51B924E9AA1BA253A1
128. 庞，G.，陆，L.，Karniadakis，G.E.：fPINNs：分数物理信息神经网络。SIAM科学计算杂志**41**(4), A2603–A2626 (2019). https://doi.org/10.1137/18M1229845,https://epubs.siam.org/doi/abs/10.1137/18M1229845
129. 帕兹克，A.，格罗斯，S.，钦塔拉，S.，等：PyTorch中的自动微分。技术报告，(2017)https://openreview.net/forum?id=BJJsrmfCZ
130. 帕特尔，R.G.，马尼克姆，I.，特拉斯克，N.A.，等：用于双曲系统的热力学一致物理信息神经网络。计算物理杂志**449**, 110,754 (2022). https://doi.org/10.1016/j.jcp.2021.110754, https://www.sciencedirect.com/science/article/pii/S0021999121006495
131. 佩德罗，J.B.，马罗纳斯，J.，帕雷德斯，R.：用神经网络求解偏微分方程。(2019)arXiv:1912.04737 [物理]
132. 彭，W.，张，J.，周，W.，等：IDRLnet：一个物理信息神经网络库。(2021)arXiv:2107.04320 [计算机科学，数学]
133. Pinkus，A.：神经网络中MLP模型的逼近理论。数值年刊**8**，143–195 (1999)。 https://doi.org/10.1017/S0962492900002919,www.cambridge.org/core/journals/acta-numerica/article/abs/approximation-theory-of-the-mlp-model-in-neural-networks/18072C558C8410C4F92A82BCC8FC8CF9
134. Pratama，D.A.，Bakar，M.A.，Man，M.，等：基于ANNs的方法求解偏微分方程：综述。(2021) 预印本 https://doi.org/10.20944/preprints202102.0160.v1,https://www.preprints.org/manuscript/202102.0160/v1
135. Psichogios，D.C。，Ungar，L.H。：混合神经网络-第一性原理方法用于过程建模。AIChE期刊**38**(10), 1499–1511 (1992)。 https://doi.org/10.1002/aic.690381003, https://onlinelibrary.wiley.com/doi/abs/10.1002/aic.690381003

136. Quarteroni, A.: Numerical Models for Differential Problems, 2nd edn. Springer Publishing Company, Incorporated (2013)
137. Rackauckas, C., Ma, Y., Martensen, J., et al.: Universal Differential Equations for Scientific Machine Learning. (2021) arXiv:2001.04385 [cs, math, q-bio, stat]
138. Rafiq, M., Rafiq, G., Choi, G.S.: DSFA-PINN: Deep Spectral Feature Aggregation Physics Informed Neural Network. IEEE Access **10**, 1 (2022). https://doi.org/10.1109/ACCESS.2022.3153056
139. Raissi, M.: Deep hidden physics models: Deep learning of nonlinear partial differential equations. J. Mach. Learn. Res. **19**(25), 1–24 (2018). http://jmlr.org/papers/v19/18-046.html
140. Raissi, M., Karniadakis, G.E.: Hidden physics models: Machine learning of nonlinear partial differential equations. J. Comput. Phys. **357**, 125–141 (2018). https://doi.org/10.1016/j.jcp.2017.11.039, www.sciencedirect.com/science/article/pii/S0021999117309014
141. Raissi, M., Perdikaris, P., Karniadakis, G.E.: Inferring solutions of differential equations using noisy multi-fidelity data. J. Comput. Phys. **335**, 736–746 (2017). https://doi.org/10.1016/j.jcp.2017.01.060, www.sciencedirect.com/science/article/pii/S0021999117300761
142. Raissi, M., Perdikaris, P., Karniadakis, G.E.: Machine learning of linear differential equations using Gaussian processes. J. Comput. Phys. **348**, 683–693 (2017). https://doi.org/10.1016/j.jcp.2017.07.050, www.sciencedirect.com/science/article/pii/S0021999117305582
143. Raissi, M., Perdikaris, P., Karniadakis, G.E.: Physics Informed Deep Learning (Part I): Data-driven Solutions of Nonlinear Partial Differential Equations. (2017c) arXiv:1711.10561 [cs, math, stat]
144. Raissi, M., Perdikaris, P., Karniadakis, G.E.: Physics Informed Deep Learning (Part II): Data-driven Discovery of Nonlinear Partial Differential Equations. (2017d) arXiv:1711.10566 [cs, math, stat]
145. Raissi, M., Perdikaris, P., Karniadakis, G.E.: Numerical gaussian processes for time-dependent and nonlinear partial differential equations. SIAM J. Sci. Comput. **40**(1), A172–A198 (2018). https://doi.org/10.1137/17M1120762
146. Raissi, M., Perdikaris, P., Karniadakis, G.E.: Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. J. Comput. Phys. **378**, 686–707 (2019). https://doi.org/10.1016/j.jcp.2018.10.045, www.sciencedirect.com/science/article/pii/S0021999118307125
147. Raissi, M., Yazdani, A., Karniadakis, G.E.: Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations. Sci. **367**(6481), 1026–1030 (2020). https://doi.org/10.1126/science.aaw4741, www.science.org/doi/10.1126/science.aaw4741
148. Ramabathiran, A.A., Ramachandran, P.: SPINN: Sparse, Physics-based, and partially Interpretable Neural Networks for PDEs. J. Comput. Phys. **445**, 110,600 (2021). https://doi.org/10.1016/j.jcp.2021.110600, https://www.sciencedirect.com/science/article/pii/S0021999121004952
149. Rudin, C., Chen, C., Chen, Z., et al.: Interpretable machine learning: Fundamental principles and 10 grand challenges. Stat. Surveys **16**(none), 1–85 (2022). https://doi.org/10.1214/21-SS133, https://projecteuclid.org/journals/statistics-surveys/volume-16/issue-none/Interpretable-machine-learning-Fundamental-principles-and-10-grand-challenges/10.1214/21-SS133.full
150. Ryaben’kii, V.S., Tsynkov, S.V.: A Theoretical Introduction to Numerical Analysis. CRC Press, Boca Raton, FL (2006)
151. Sahli Costabal, F., Yang, Y., Perdikaris, P., et al.: Physics-Informed Neural Networks for Cardiac Activation Mapping. Front. Phys. **8**, 42 (2020). https://doi.org/10.3389/fphy.2020.00042, www.frontiersin.org/article/10.3389/fphy.2020.00042
152. Scharzenberger, C., Hays, J.: Learning To Estimate Regions Of Attraction Of Autonomous Dynamical Systems Using Physics-Informed Neural Networks. Tech. rep., (2021) https://doi.org/10.48550/arXiv.2111.09930, arXiv:2111.09930 [cs] type: article
153. Schiassi, E., Furfaro, R., Leake, C., et al.: Extreme theory of functional connections: A fast physics-informed neural network method for solving ordinary and partial differential equations. Neurocomputing **457**, 334–356 (2021). https://doi.org/10.1016/j.neucom.2021.06.015, www.sciencedirect.com/science/article/pii/S0925231221009140
154. Schölkopf, B., Locatello, F., Bauer, S., et al.: Toward Causal Representation Learning. Proc. IEEE **109**(5), 612–634 (2021). https://doi.org/10.1109/JPROC.2021.3058954
155. Sengupta, S., Basak, S., Saikia, P., et al.: A review of deep learning with special emphasis on architectures, applications and recent trends. Knowledge-Based Systems **194**, 105,596 (2020). https://doi.org/10.1016/j.knosys.2020.105596, https://www.sciencedirect.com/science/article/pii/S095070512030071X
156. Sergeev, A., Del Balso, M.: Horovod: fast and easy distributed deep learning in TensorFlow. Tech. rep., (2018) https://doi.org/10.48550/arXiv.1802.05799, arXiv:1802.05799 [cs, stat] type: article
157. Shin, Y., Darbon, J., Karniadakis, G.E.: On the convergence of physics informed neural networks for linear second-order elliptic and parabolic type PDEs. Commun. Comput. Phys. **28**(5), 2042–2074 (2020a). https://doi.org/10.4208/cicp.OA-2020-0193, arXiv: 2004.01806

136. Quarteroni, A.: Numerical Models for Differential Problems, 2nd edn. Springer Publishing Company, Incorporated (2013)
137. Rackauckas, C., Ma, Y., Martensen, J., et al.: Universal Differential Equations for Scientific Machine Learning. (2021) arXiv:2001.04385 [cs, math, q-bio, stat]
138. 拉菲克，M.，拉菲克，G.，崔，G.S.：DSFA-PINN：深度频谱特征聚合物理信息神经网络。IEEE Access **10**， 1 (2022).https://doi.org/10.1109/ACCESS.2022.3153056
139. 拉西，M.：深度隐藏物理模型：非线性偏微分方程的深度学习。机器学习研究杂志 **19**(25),1–24 (2018). http://jmlr.org/papers/v19/18-046.html
140. 拉西，M.，Karniadakis，G.E.：隐藏物理模型：非线性偏微分方程的机器学习。计算物理杂志 **357**， 125–141(2018). https://doi.org/10.1016/j.jcp.2017.11.039， www.sciencedirect.com/science/article/pii/S0021999117309014
141. 拉西，M.，佩德里卡里斯，P.，卡尼亚迪亚斯，G.E.：使用噪声多保真数据推断微分方程的解。计算物理杂志 **335**, 736–746 (2017).https://doi.org/10.1016/j.jcp.2017.01.060,www.sciencedirect.com/science/article/pii/S0021999117300761
142. 拉西，M.，佩德里卡里斯，P.，卡尼亚迪亚斯，G.E.：使用高斯过程进行线性微分方程的机器学习。计算物理杂志 **348**, 683–693 (2017). https://doi.org/10.1016/j.jcp.2017.07.050,www.sciencedirect.com/science/article/pii/S0021999117305582
143. 拉西，M.，佩德里卡里斯，P.，卡尼亚迪亚斯，G.E.：物理信息深度学习（第一部分）：非线性偏微分方程的数据驱动解。(2017c)arXiv:1711.10561 [cs, math, stat]
144. 拉西，M.，佩德里卡里斯，P.，卡尼亚迪亚斯，G.E.：物理信息深度学习（第二部分）：数据驱动的非线性偏微分方程发现。(2017d) arXiv:1711.10566 [cs, math, stat]
145. 拉西，M.，佩德里卡里斯，P.，卡尼亚迪亚斯，G.E.：时变和非线性偏微分方程的数值高斯过程。SIAM科学计算杂志 **40**(1), A172–A198(2018). https://doi.org/10.1137/17M1120762
146. 拉西，M.，佩德里卡里斯，P.，卡尼亚迪亚斯，G.E.：物理信息神经网络：一个深度学习框架，用于解决涉及非线性偏微分方程的正向和逆问题。计算物理杂志 **378**， 686–707 (2019). https://doi.org/10.1016/j.jcp.2018.10.045,www.sciencedirect.com/science/article/pii/S0021999118307125
147. 拉西，M.，亚兹达尼，A.，卡尼亚达基斯，G.E.：隐藏流体力学：从流动可视化中学习速度和压力场。Sci. **367**(6481), 1026–1030 (2020)。 https://doi.org/10.1126/science.aaw4741， www.science.org/doi/10.1126/science.aaw4741
148. 拉马巴西兰，A.A.，拉马钱德拉南，P.：SPINN：稀疏、基于物理的、部分可解释的神经网络用于偏微分方程。计算物理杂志 **445**, 110,600 (2021)。 https://doi.org/10.1016/j.jcp.2021.110600， https://www.sciencedirect.com/science/article/pii/S0021999121004952
149. 鲁丁，C.，陈，C.，陈，Z.，等：可解释机器学习：基本原理和10个重大挑战。统计调查 **16**(无)，1–85 (2022)。 https://doi.org/10.1214/21-SS133, https://projecteuclid.org/journals/statistics-surveys/volume-16/issue-none/Interpretable-machine-learning-Fundamental-principles-and-10-grand-challenges/10.1214/21-SS133.full
150. Ryaben’kii, V.S., Tsynkov, S.V.: 数值分析的理论入门. CRC出版社, Boca Raton,FL (2006)
151. Sahli Costabal, F., Yang, Y., Perdikaris, P., 等: 物理信息神经网络用于心脏活动映射. 物理前沿 **8**, 42 (2020). https://doi.org/10.3389/fphy.2020.00042,www.frontiersin.org/article/10.3389/fphy.2020.00042
152. Scharzenberger, C., Hays, J.: 使用物理信息神经网络学习估计自主动力系统的吸引区域. 技术报告, (2021) https://doi.org/10.48550/arXiv.2111.09930,arXiv:2111.09930 [cs] 类型: 文章
153. Schiassi, E., Furfaro, R., Leake, C., 等: 功能连接的极端理论: 一种快速物理信息神经网络方法用于求解常微分方程和偏微分方程. 神经计算**457**, 334–356 (2021). https://doi.org/10.1016/j.neucom.2021.06.015,www.sciencedirect.com/science/article/pii/S0925231221009140
154. Schölkopf, B., Locatello, F., Bauer, S., 等：因果表示学习。IEEE会议录109卷第5期，612–634页（2021）。**109**(5),612–634 (2021).https://doi.org/10.1109/JPROC.2021.3058954
155. Sengupta, S., Basak, S., Saikia, P., 等：深度学习综述，特别强调架构、应用和最新趋势。知识系统194卷，105,596页（2020）。**194**, 105,596 (2020). https://doi.org/10.1016/j.knosys.2020.105596, https://www.sciencedirect.com/science/article/pii/S095070512030071X
156. Sergeev, A., Del Balso, M.: Horovod：在TensorFlow中快速且易于使用的分布式深度学习。技术报告，（2018) https://doi.org/10.48550/arXiv.1802.05799,arXiv:1802.05799 [cs, stat] 类型：文章
157. Shin, Y., Darbon, J., Karniadakis, G.E.: 关于物理信息神经网络在一线性二阶椭圆型和抛物型偏微分方程收敛性研究. 计算物理通讯 **28**(5), 2042–2074(2020a).https://doi.org/10.4208/cicp.OA-2020-0193,arXiv: 2004.01806

158. Shin, Y., Zhang, Z., Karniadakis, G.E.: Error estimates of residual minimization using neural networks for linear PDEs. (2020b) arXiv:2010.08019 [cs, math]
159. Shrestha, A., Mahmood, A.: Review of Deep Learning Algorithms and Architectures. IEEE Access **7**, 53040–53065 (2019). https://doi.org/10.1109/ACCESS.2019.2912200
160. Sirignano, J., Spiliopoulos, K.: DGM: A deep learning algorithm for solving partial differential equations. J. Comput. Phys. **375**, 1339–1364 (2018). https://doi.org/10.1016/j.jcp.2018.08.029, www.sciencedirect.com/science/article/pii/S0021999118305527
161. Sitzmann, V., Martel, J.N.P., Bergman, A.W., et al.: Implicit Neural Representations with Periodic Activation Functions (2020). arXiv:2006.09661 [cs, eess]
162. Smith, J.D., Azizzadenesheli, K., Ross, Z.E.: EikoNet: Solving the Eikonal Equation With Deep Neural Networks. IEEE Trans. Geosci. Remote Sens. **59**(12), 10685–10696 (2021). https://doi.org/10.1109/TGRS.2020.3039165
163. Smith, J.D., Ross, Z.E., Azizzadenesheli, K., et al.: HypoSVI: Hypocentre inversion with Stein variational inference and physics informed neural networks. Geophys. J. Int. **228**(1), 698–710 (2021). https://doi.org/10.1093/gji/ggab309
164. Stein, M.L.: Large sample properties of simulations using latin hypercube sampling. Technometrics **29**, 143–151 (1987)
165. Stiasny J, Misyris GS, Chatzivasileiadis S (2021) Physics-Informed Neural Networks for Non-linear System Identification for Power System Dynamics. In: 2021 IEEE Madrid PowerTech, pp 1–6, https://doi.org/10.1109/PowerTech46648.2021.9495063
166. Stielow, T., Scheel, S.: Reconstruction of nanoscale particles from single-shot wide-angle free-electron-laser diffraction patterns with physics-informed neural networks. Phys. Review E **103**(5), 053,312 (2021). https://doi.org/10.1103/PhysRevE.103.053312, https://link.aps.org/doi/10.1103/PhysRevE.103.053312
167. Stiller, P., Bethke, F., Böhme, M., et al.: Large-Scale Neural Solvers for Partial Differential Equations. In: Nichols J, Verastegui B, Maccabe AB, et al (eds) Driving Scientific and Engineering Discoveries Through the Convergence of HPC, Big Data and AI. Springer International Publishing, Cham, Communications in Computer and Information Science, pp. 20–34, (2020) https://doi.org/10.1007/978-3-030-63393-6_2
168. Sun, L., Gao, H., Pan, S., et al.: Surrogate modeling for fluid flows based on physics-constrained deep learning without simulation data. Comput. Methods Appl. Mech. Engrg. **361**, 112,732 (2020a). https://doi.org/10.1016/j.cma.2019.112732, https://www.sciencedirect.com/science/article/pii/S004578251930622X
169. Sun, S., Cao, Z., Zhu, H., et al.: A Survey of Optimization Methods From a Machine Learning Perspective. IEEE Trans. Cybernet. **50**(8), 3668–3681 (2020). https://doi.org/10.1109/TCYB.2019.2950779
170. Tartakovsky, A.M., Marrero, C.O., Perdikaris, P., et al.: Physics-Informed Deep Neural Networks for Learning Parameters and Constitutive Relationships in Subsurface Flow Problems. Water Resources Research **56**(5), e2019WR026,731 (2020). https://doi.org/10.1029/2019WR026731, https://onlinelibrary.wiley.com/doi/abs/10.1029/2019WR026731
171. Thompson, D.B.: Numerical Methods 101 – Convergence of Numerical Models. ASCE, pp. 398–403 (1992), https://cedb.asce.org/CEDBsearch/record.jsp?dockey=0078142
172. Tong, Y., Xiong, S., He, X., et al.: Symplectic neural networks in taylor series form for hamiltonian systems. J. Comput. Phys. **437**, 110,325 (2021). https://doi.org/10.1016/j.jcp.2021.110325, https://www.sciencedirect.com/science/article/pii/S0021999121002205
173. Torabi Rad, M., Viardin, A., Schmitz, G.J., et al.: Theory-training deep neural networks for an alloy solidification benchmark problem. Comput. Mater. Sci. **180**, 109,687 (2020). https://doi.org/10.1016/j.commatsci.2020.109687, https://www.sciencedirect.com/science/article/pii/S0927025620301786
174. Viana, F.A.C., Nascimento, R.G., Dourado, A., et al.: Estimating model inadequacy in ordinary differential equations with physics-informed neural networks. Comput. Structures **245**, 106,458 (2021). https://doi.org/10.1016/j.compstruc.2020.106458, https://www.sciencedirect.com/science/article/pii/S0045794920302613
175. Waheed, U.b., Haghighat, E., Alkhalifah, T., et al.: PINNeik: Eikonal solution using physics-informed neural networks. Comput. Geosci. **155**, 104,833 (2021). https://doi.org/10.1016/j.cageo.2021.104833, https://www.sciencedirect.com/science/article/pii/S009830042100131X
176. Wang, L., Yan, Z.: Data-driven rogue waves and parameter discovery in the defocusing nonlinear Schrödinger equation with a potential using the PINN deep learning. Phys. Lett. A **404**, 127,408 (2021). https://doi.org/10.1016/j.physleta.2021.127408, https://www.sciencedirect.com/science/article/pii/S0375960121002723
177. Wang, N., Chang, H., Zhang, D.: Theory-guided auto-encoder for surrogate construction and inverse modeling. Comput. Methods Appl. Mech. Engrg. **385**, 114,037 (2021a). https://doi.org/10.1016/j.cma.2021.114037, https://www.sciencedirect.com/science/article/pii/S0045782521003686

158. Shin, Y., Zhang,Z., Karniadakis, G.E.: 使用神经网络进行线性偏微分方程的残差最小化的误差估计。(2020b) arXiv:2010.08019 [cs, math]Shrestha, A., Mahmood, A.: 深度学习算法和架构的综述。IEEE Access **7**,
159.
53040–53065 (2019). https://doi.org/10.1109/ACCESS.2019.2912200
160. Sirignano, J., Spiliopoulos, K.: DGM：一种用于求解偏微分方程的深度学习算法。计算物理杂志 **375**， 1339–1364(2018)。 https://doi.org/10.1016/j.jcp.2018.08.029， www.sciencedirect.com/science/article/pii/S0021999118305527
161. Sitzmann, V., Martel, J.N.P., Bergman, A.W., 等：具有周期性激活函数的隐式神经表示 (2020)。arXiv:2006.09661 [cs, eess]
162. Smith, J.D., Azizzadenesheli, K., Ross, Z.E.: EikoNet: 使用深度神经网络求解Eikonal方程。IEEE Trans. Geosci. Remote Sens. **59**(12), 10685–10696(2021). https://doi.org/10.1109/TGRS.2020.3039165
163. Smith, J.D., Ross, Z.E., Azizzadenesheli, K., 等：HypoSVI：使用斯坦变分推理和物理信息神经网络进行震中反演。Geophys. J. Int. **228**(1), 698–710(2021). https://doi.org/10.1093/gji/ggab309
164. Stein, M.L.: 使用拉丁超立方采样的模拟的大样本性质. Technometrics **29**, 143–151 (1987)
165. Stiasny J, Misyris GS, Chatzivasileiadis S (2021) 用于电力系统动力学非线性系统识别的物理信息神经网络. In: 2021IEEE Madrid PowerTech, pp 1–6, https://doi.org/10.1109/PowerTech46648.2021.9495063
166. Stielow, T., Scheel, S.: 从单次宽角自由电子激光衍射图中重建纳米级颗粒——基于物理信息神经网络. 物理评论E**103**(5), 053,312 (2021).
https://doi.org/10.1103/PhysRevE.103.053312,https://link.aps.org/doi/10.1103/PhysRevE.103.053312
167. Stiller, P., Bethke, F., Böhme, M., 等: 用于偏微分方程的大规模神经求解器. In: NicholsJ, Verastegui B, Maccabe AB, 等 (eds) 通过HPC、大数据和AI的融合推动科学与工程发现. 斯普林格国际出版公司, 查姆, 计算机与信息科学通讯, pp. 20–34, (2020) https://doi.org/10.1007/978-3-030-63393-6_2
168. 孙，L.，高，H.，潘，S.，等：基于物理约束的深度学习流体流动代理建模，无需模拟数据。计算方法与应用力学工程**361**，112，732(2020a). https://doi.org/10.1016/j.cma.2019.112732,
https://www.sciencedirect.com/science/article/pii/S004578251930622X
169. 孙，S.，曹，Z.，朱，H.，等：从机器学习视角看优化方法综述。IEEE 控制论汇刊**50**(8)，3668–3681 (2020).https://doi.org/10.1109/TCYB.2019.2950779
170. 塔塔科夫斯基，A.M.，马拉罗，C.O.，佩德里卡里斯，P.，等：物理信息深度融合神经网络用于地下流动问题参数与本构关系学习。水资源研究**56**(5)，e2019WR026，731 (2020).
https://doi.org/10.1029/2019WR026731,https://onlinelibrary.wiley.com/doi/abs/10.1029/2019WR026731
171. 汤普森，D.B.：数值方法101——数值模型的收敛性。美国土木工程师协会，第398-403页（1992年），
https://cedb.asce.org/CEDBsearch/record.jsp?dockey=0078142
172. 童，Y.，熊，S.，何，X.，等：哈密顿系统的辛神经网络在泰勒级数形式中的应用。计算物理杂志**437**，110,325（2021年）。
https://doi.org/10.1016/j.jcp.2021.110325，https://www.sciencedirect.com/science/article/pii/S0021999121002205
173. 托拉比·拉德，M.，维达林，A.，施密茨，G.J.，等：用于合金凝固基准问题的理论训练深度神经网络。计算材料科学**180**，109,687(2020年)。https://doi.org/10.1016/j.commatsci.2020.109687，https://www.sciencedirect.co
m/science/article/pii/S0927025620301786
174. 维亚纳，F.A.C.，纳西メント，R.G.，多拉多，A.，等：使用物理信息神经网络估计常微分方程中的模型不充分性。计算结构 **245**，106,458 (2021)。
https://doi.org/10.1016/j.compstruc.2020.106458,https://www.sciencedirect.com/science/article/pii/S0045794920302613
175. 瓦希德，U.b.，哈吉加特，E.，阿尔哈利法，T.，等：PINNeik：使用物理信息神经网络的特征解。计算地球科学 **155**，104,833 (2021)。 https://doi.org/10.1016/j.cageo.2021.104833,https://www.sciencedirect.com/science/article/pii/S009830042100131X
176. 王，L.，严，Z.：使用PINN深度学习在具有势的聚焦非线性薛定谔方程中进行数据驱动的 roguewaves 和参数发现。物理快报A**404**，127,408 (2021)。
https://doi.org/10.1016/j.physleta.2021.127408,https://www.sciencedirect.com/science/article/pii/S0375960121002723
177. 王，N.，张，H.，张，D.：理论指导自动编码器用于代理构建和逆向建模。计算方法与应用力学工程**385**，114,037 (2021a)。https://doi.org/10.1016/j.cma.2021.114037,
https://www.sciencedirect.com/science/article/pii/S0045782521003686

178. Wang, S., Perdikaris, P.: Deep learning of free boundary and Stefan problems. J. Comput. Phys. **428**, 109,914 (2021). https://doi.org/10.1016/j.jcp.2020.109914, https://www.sciencedirect.com/science/article/pii/S0021999120306884
179. Wang, S., Teng, Y., Perdikaris, P.: Understanding and Mitigating Gradient Flow Pathologies in Physics-Informed Neural Networks. SIAM J. Sci. Comput. **43**(5), A3055–A3081 (2021). https://doi.org/10.1137/20M1318043, https://epubs.siam.org/doi/abs/10.1137/20M1318043
180. Wang, S., Sankaran, S., Perdikaris, P.: Respecting causality is all you need for training physics-informed neural networks. (2022a) arXiv:2203.07404 [nlin, physics:physics, stat]
181. Wang, S., Yu, X., Perdikaris, P.: When and why PINNs fail to train: A neural tangent kernel perspective. J. Comput. Phys. **449**, 110,768 (2022b). https://doi.org/10.1016/j.jcp.2021.110768, https://www.sciencedirect.com/science/article/pii/S002199912100663X
182. Wen, G., Li, Z., Azizzadenesheli, K., et al.: U-FNO–An enhanced Fourier neural operator-based deep-learning model for multiphase flow. Adv. Water Resources **163**, 104,180 (2022). https://doi.org/10.1016/j.advwatres.2022.104180, https://www.sciencedirect.com/science/article/pii/S0309170822000562
183. Wiecha, P.R., Arbouet, A., Arbouet, A., et al.: Deep learning in nano-photonics: inverse design and beyond. Photonics Research **9**(5), B182–B200 (2021). https://doi.org/10.1364/PRJ.415960, www.osapublishing.org/prj/abstract.cfm?uri=prj-9-5-B182
184. Wong, J.C., Gupta, A., Ong, Y.S.: Can Transfer Neuroevolution Tractably Solve Your Differential Equations? IEEE Comput. Intell. Mag. **16**(2), 14–30 (2021). https://doi.org/10.1109/MCI.2021.3061854
185. Wong, J.C., Ooi, C., Gupta, A., et al.: Learning in Sinusoidal Spaces with Physics-Informed Neural Networks. (2022) arXiv:2109.09338 [physics]
186. Xiao, H., Wu, J.L., Laizet, S., et al.: Flows over periodic hills of parameterized geometries: A dataset for data-driven turbulence modeling from direct simulations. Comput. Fluids **200**, 104,431 (2020). https://doi.org/10.1016/j.compfluid.2020.104431, https://www.sciencedirect.com/science/article/pii/S0045793020300074
187. Xu, K., Darve, E.: ADCME: Learning Spatially-varying Physical Fields using Deep Neural Networks. (2020) arXiv:2011.11955 [cs, math]
188. Xu, K., Darve, E.: Solving inverse problems in stochastic models using deep neural networks and adversarial training. Comput. Methods Appl. Mech. Engrg. **384**, 113,976 (2021). https://doi.org/10.1016/j.cma.2021.113976, https://www.sciencedirect.com/science/article/pii/S0045782521003078
189. Yang, L., Zhang, D., Karniadakis, G.E.: Physics-informed generative adversarial networks for stochastic differential equations. SIAM J. Sci. Comput. **42**(1), A292–A317 (2020). https://doi.org/10.1137/18M1225409
190. Yang, L., Meng, X., Karniadakis, G.E.: B-PINNs: Bayesian physics-informed neural networks for forward and inverse PDE problems with noisy data. J. Comput. Phys. **425**, 109,913 (2021). https://doi.org/10.1016/j.jcp.2020.109913, https://www.sciencedirect.com/science/article/pii/S0021999120306872
191. Yang, Y., Perdikaris, P.: Adversarial uncertainty quantification in physics-informed neural networks. J. Comput. Phys. **394**, 136–152 (2019). https://doi.org/10.1016/j.jcp.2019.05.027, www.sciencedirect.com/science/article/pii/S0021999119303584
192. Yarotsky, D.: Error bounds for approximations with deep relu networks. Neural Netw. **94**, 103–114 (2017). https://doi.org/10.1016/j.neunet.2017.07.002, www.sciencedirect.com/science/article/pii/S0893608017301545
193. Yuan, L., Ni, Y.Q., Deng, X.Y., et al.: A-PINN: Auxiliary physics informed neural networks for forward and inverse problems of nonlinear integro-differential equations. J. Comput. Phys. **462**, 111,260 (2022). https://doi.org/10.1016/j.jcp.2022.111260,https://www.sciencedirect.com/science/article/pii/S0021999122003229
194. Yucesan, Y.A., Viana, F.A.C.: Hybrid physics-informed neural networks for main bearing fatigue prognosis with visual grease inspection. Comput. Ind. **125**:103,386 (2021). https://doi.org/10.1016/j.compind.2020.103386, https://www.sciencedirect.com/science/article/pii/S0166361520306205
195. Zhang, D., Lu, L., Guo, L., et al.: Quantifying total uncertainty in physics-informed neural networks for solving forward and inverse stochastic problems. J. Comput. Phys. **397**, 108,850 (2019). https://doi.org/10.1016/j.jcp.2019.07.048, https://www.sciencedirect.com/science/article/pii/S0021999119305340
196. Zhang, D., Guo, L., Karniadakis, G.E.: Learning in modal space: Solving time-dependent stochastic pdes using physics-informed neural networks. SIAM J. Sci. Comput. **42**(2), A639–A665 (2020). https://doi.org/10.1137/19M1260141
197. Zhang, R., Liu, Y., Sun, H.: Physics-informed multi-LSTM networks for metamodeling of nonlinear structures. Comput. Methods Appl. Mech. Engrg. **369**, 113,226 (2020b). https://doi.org/10.1016/j.cma.2020.113226, https://www.sciencedirect.com/science/article/pii/S0045782520304114

178. 王，S., 佩德里卡里斯，P.: 深度学习自由边界和Stefan问题。计算物理杂志 **428**， 109,914 (2021)。 https://doi.org/10.1016/j.jcp.2020.109914,https://www.sciencedirect.com/science/article/pii/S0021999120306884
179. 王，S., 腾，Y., 佩德里卡里斯，P.: 理解和缓解物理信息神经网络中的梯度流病态。SIAM科学计算杂志 **43**(5), A3055–A3081 (2021)。 https://doi.org/10.1137/20M1318043, https://epubs.siam.org/doi/abs/10.1137/20M1318043
180. 王，S., 桑卡拉南，S., 佩德里卡里斯，P.: 尊重因果性是训练物理信息神经网络所需的一切。(2022a) arXiv:2203.07404 [nlin, physics:physics, stat]
181. 王，S., 余，X., 佩德里卡里斯，P.: 何时以及为何物理信息神经网络（PINNs）无法训练：基于神经切线核的视角。计算物理杂志 **449**, 110,768 (2022b). https://doi.org/10.1016/j.jcp.2021.110768, https://www.sciencedirect.com/science/article/pii/S002199912100663X
182. 温，G., 李，Z., Azizzadenesheli, K., 等：U-FNO——一种基于傅里叶神经算子（FNO）的深度学习模型，用于多相流。先进水资源杂志 **163**, 104,180 (2022). https://doi.org/10.1016/j.advwatres.2022.104180, https://www.sciencedirect.com/science/article/pii/S0309170822000562
183. Wiecha, P.R., Arbouet, A., Arbouet, A., 等：纳光子学中的深度学习：逆向设计与展望。光子学研究 **9**(5), B182–B200(2021). https://doi.org/10.1364/PRJ.415960, www.osapublishing.org/prj/abstract.cfm?uri=prj-9-5-B182
184. 王, J.C.， Gupta, A.， Ong, Y.S.： 迁移神经进化能否可扩展地解决您的常微分方程？IEEE计算智能杂志**16**(2)， 14–30 (2021)。 https://doi.org/10.1109/MCI.2021.3061854
185. 王, J.C.， 黄, C.， Gupta, A.， 等： 使用物理信息神经网络的正弦空间学习。(2022)arXiv:2109.09338 [physics]
186. 小浩， H.， 吴， J.L.， 拉泽特， S.， 等： 参数化几何形状的周期性山丘上的流动： 从直接模拟中获取数据驱动湍流建模数据集。计算流体**200**， 104,431 (2020)。 https://doi.org/10.1016/j.compfluid.2020.104431,https://www.sciencedirect.com/science/article/pii/S0045793020300074徐，K.， Darve,E.： ADCME： 使用深度神经网络学习空间变化的物理场。
187.

(2020) arXiv:2011.11955 [cs, math]
188. Xu, K., Darve, E.: 使用深度神经网络和对抗训练求解随机模型中的逆问题。计算方法与应用力学工程 **384**, 113,976 (2021).https://doi.org/10.1016/j.cma.2021.113976, https://www.sciencedirect.com/science/article/pii/S0045782521003078
189. Yang, L., Zhang, D., Karniadakis, G.E.: 物理信息生成对抗网络用于随机微分方程。SIAM科学计算杂志 **42**(1), A292–A317 (2020).https://doi.org/10.1137/18M1225409
190. Yang, L., Meng, X., Karniadakis, G.E.: B-PINNs：贝叶斯物理信息神经网络用于含噪声数据的正向和逆偏微分方程问题。计算物理杂志 **425**, 109,913 (2021).https://doi.org/10.1016/j.jcp.2020.109913, https://www.sciencedirect.com/science/article/pii/S0021999120306872
191. Yang, Y., Perdikaris, P.: 对抗不确定性量化在物理信息神经网络中 .
计算物理杂志<sup>394</sup>, 136–152 (2019). <sup>https://doi.org/10.1016/j.jcp.2019.05.027</sup>, <sup>www.sciencedirect.com/science/article/pii/S0021999119303584</sup>
192. Yarotsky, D.: 深度ReLU网络逼近的误差界限。神经网络 **94**, 103–114 (2017). https://doi.org/10.1016/j.neunet.2017.07.002,www.sciencedirect.com/science/article/pii/S0893608017301545
193. Yuan, L., Ni, Y.Q., Deng, X.Y., et al.: A-PINN：用于非线性积分微分方程正向和逆问题的辅助物理信息神经网络。计算物理杂志 **462**, 111,260(2022). https://doi.org/10.1016/j.jcp.2022.111260,https://www.sciencedirect.com/science/article/pii/S0021999122003229
194. 尤塞桑，Y.A.，维亚纳，F.A.C.：基于混合物理信息神经网络的视觉润滑脂检查主轴承疲劳预测。计算工业**125**：103,386 (2021)。 https://doi.org/10.1016/j.compind.2020.103386， https://www.sciencedirect.com/science/article/pii/S0166361520306205
195. 张，D.，陆，L.，郭，L.，等：用于解决正向和逆随机问题的物理信息神经网络的总体不确定性量化。计算物理杂志**397**， 108,850 (2019)。https://doi.org/10.1016/j.jcp.2019.07.048， https://www.sciencedirect.com/science/article/pii/S0021999119305340
196. 张，D.，郭，L.，Karniadakis，G.E.：模态空间中的学习：使用物理信息神经网络解决时变随机偏微分方程。SIAM科学计算杂志**42**(2)， A639–A665 (2020)。 https://doi.org/10.1137/19M1260141
197. 张，R.，刘，Y.，孙，H.：物理信息融合多长短期记忆网络用于非线性结构的元建模。计算方法与应用力学工程**369**, 113,226 (2020b).https://doi.org/10.1016/j.cma.2020.113226, https://www.sciencedirect.com/science/article/pii/S0045782520304114

198. Zhi-Qin, Xu, J., et al.: Frequency principle: Fourier analysis sheds light on deep neural networks. Commun. Comput. Phys. **28**(5), 1746–1767 (2020). https://doi.org/10.4208/cicp.OA-2020-0085, http://global-sci.org/intro/article_detail/cicp/18395.html
199. Zhu, Q., Liu, Z., Yan, J.: Machine learning for metal additive manufacturing: predicting temperature and melt pool fluid dynamics using physics-informed neural networks. Comput. Mech. **67**(2), 619–635 (2021). https://doi.org/10.1007/s00466-020-01952-9
200. Zhu, Y., Zabaras, N., Koutsourelakis, P.S., et al.: Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data. J. Comput. Phys. **394**, 56–81 (2019). https://doi.org/10.1016/j.jcp.2019.05.024, www.sciencedirect.com/science/article/pii/S0021999119303559
201. Zubov, K., McCarthy, Z., Ma, Y., et al.: NeuralPDE: Automating Physics-Informed Neural Networks (PINNs) with Error Approximations. (2021a) arXiv:2107.09443 [cs]
202. Zubov, K., McCarthy, Z., Ma, Y., et al.: NeuralPDE: Automating Physics-Informed Neural Networks (PINNs) with Error Approximations. (2021b) arXiv:2107.09443 [cs]

Authors and Affiliations

Salvatore Cuomo[1] · Vincenzo Schiano Di Cola[2] · Fabio Giampaolo[1] · Gianluigi Rozza[3] · Maziar Raissi[4] · Francesco Piccialli[1]

Salvatore Cuomo
salvatore.cuomo@unina.it

Vincenzo Schiano Di Cola
vincenzo.schianodicola@unina.it

Fabio Giampaolo
fabio.giampaolo@unina.it

Gianluigi Rozza
grozza@sissa.it

Maziar Raissi
mara4513@colorado.edu

1 Department of Mathematics and Applications “Renato Caccioppoli”, University of Naples Federico II, Napoli 80126, Italy

2 Department of Electrical Engineering and Information Technology, University of Naples Federico II, Via Claudio, Napoli 80125, Italy

3 SISSA, International School for Advanced Studies, Mathematics Area, mathLab, via Bonomea 265, Trieste 34136, Italy

4 Department of Applied Mathematics, University of Colorado Boulder, Boulder, CO 80309, USA

198. 朱志琴，徐，J.，等：频率原理：傅里叶分析揭示深度神经网络。计算物理通讯 **28**(5), 1746–1767(2020). https://doi.org/10.4208/cicp.OA-2020-0085, http://global-sci.org/intro/article_detail/cicp/18395.html.

199 朱，Q.，刘，Z.，严，J.：机器学习用于金属增材制造：利用物理信息神经网络预测温度和熔池流体动力学。计算力学 **67**(2), 619–635(2021). https://doi.org/10.1007/s00466-020-01952-9

200. 朱,Y.，扎巴拉斯，N.，库斯托雷拉科斯，P.S.，等：物理约束深度学习用于高维代理建模和不带标记数据的不确定性量化。计算物理杂志 **394**,56–81 (2019). https://doi.org/10.1016/j.jcp.2019.05.024,www.sciencedirect.com/science/article/pii/S0021999119303559

201. 祖博夫，K.，麦卡锡，Z.，马，Y.，等：NeuralPDE：自动化物理信息神经网络（PINNs）与误差近似。(2021a)arXiv:2107.09443 [cs]

202. 祖博夫，K.，麦卡锡，Z.，马，Y.，等：NeuralPDE：自动化物理信息神经网络 (PINNs)与误差近似。(2021b)arXiv:2107.09443 [cs]

作者与机构隶属关系

萨尔瓦托雷·库莫[1] ·文森佐·斯卡尼奥·迪科拉[2] ·法比奥·吉安帕奥洛[1] ·吉安卢基·罗扎[3] ·马齐尔·拉西[4] ·弗朗切斯科·皮奇亚利1

萨尔瓦托雷·库莫
salvatore.cuomo@unina.it 文森佐·斯
卡尼奥·迪科拉
vincenzo.schianodicola@unina.it 法
比奥·吉安帕奥洛
fabio.giampaolo@unina.it

吉安卢基·罗扎
grozza@sissa.it

马齐尔·拉西
mara4513@co 科罗拉多州

1 数学与应用系 “雷纳托·卡乔波利”，那不勒斯费德里科二世大学，那不勒斯 80126，意大利

2 电气工程与信息技术系，那不勒斯费德里科二世大学，克劳迪奥大道，那不勒斯 80125，意大利

3 国际高等研究院，国际高等研究院数学领域，数学实验室，博诺梅亚大道265号，的里雅斯特34136，意大利

4 应用数学系，科罗拉多大学博尔德分校，博尔德，科罗拉多州80309，美国