基于物理信息神经网络的非线性微分方程求解模型研究

葛淼彦[1] 李凯锋[2]
GE Miaoyan LI Kaifeng

摘 要 微分方程在工学领域广泛应用，但传统数值方法在非线性问题时存在效率低、适应性弱的局限。物理信息神经网络通过将微分方程约束嵌入损失函数，实现数据驱动的无网格化求解。文章以非线性微分方程为例并结合数值实验，探讨 PINN 在非线性微分方程求解中的优化。研究通过对比 tanh、ReLU 等激活函数及不同网络参数（深度、神经元数量、学习率）对模型性能的影响，验证 PINN 的有效性。研究表明，PINN 框架在非线性微分方程求解中具有良好的泛化能力，为复杂微分方程求解提供了新的技术路径；tanh 激活函数在导数传播中表现更优，网络深度超过 5 层后性能提升有限，学习率在 [0.000 1, 0.001] 区间可平衡收敛速度与稳定性。

关键词 物理信息神经网络；非线性微分方程；多目标损失函数

doi：10.3969/j.issn.1672-9528.2025.10.028

0 引言

微分方程作为描述自然规律和工程现象的数学工具，在流体力学、量子力学、气候建模等领域应用甚广。然而在求解实际问题中的微分方程时，常常需要面对高阶微分、非线性、尺度多等复杂情况。传统数值解法的代表包括有限差分法、有限元法等，这类方法实际应用时依赖于网格划分，在面对维度高、边界复杂或非线性强的问题时，也不免存在计算效率低下、适应性弱的局限。因此探索新的求解方法，提高求解效率和精度，逐渐成为当前微分方程研究领域的重要课题。

随着深度学习在各领域的广泛应用[1]，物理信息神经网络（physics-informed neural networks, PINN）应运而生。PINN 通过将微分方程约束嵌入神经网络的损失函数，实现了数据驱动的无网格化求解。PINN 的概念最早由 Raissi 等人[2]于 2019 年提出，通过将数据驱动和物理模型相结合，充分利用已知的物理规律和有限的数据信息，并成功应用于非线性偏微分方程的正问题和反问题求解。随后，Lu 等人[3]开发的 DeepXDE 库通过自适应残差优化提升了 PINN 的训练效率。Wang 等人[4]提出 NAS-PINN，结合神经网络架构搜索技术优化模型结构。此外，李晓峰等人[5]针对海洋孤立波问题构建了 PIRBF 模型，通过径向基函数改进 PINN 的非线性拟合能力。

然而，随着对微分方程的研究日渐深入，PINN 的不足日渐显现，求解复杂问题时面临梯度消失、收敛速度慢等挑战。现有研究对非线性微分方程的探索不足，尽管 PINN 在普通方程中表现优异，非线性微分方程的导数约束会导致损失函数复杂化，加剧模型训练的不稳定性，限制了其实际应用。本文针对非线性微分方程，结合非线性微分方程的数值实验，探究网络优化方法。

1 物理信息神经网络原理

PINN 是一种将物理定律嵌入神经网络训练的深度学习框架，通过数据驱动并且结合物理约束，解决非线性偏微分方程的正向求解和逆向参数识别问题，其基本结构如图 1 所示。

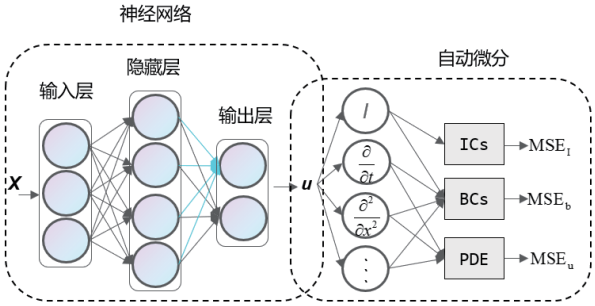

图 1 PINN 网络结构

图 1 中左侧虚线框内是一般的神经网络，其基本结构一般包括输入层、隐藏层、输出层。输入数据 X 在经过预处理后进入输入层，随后在隐藏层中计算激活，最终经输出层处理后输出为结果 u。神经网络的每个层都包含多个神经元，

1. 常州大学机械与轨道交通学院 江苏常州 2131642
2. 南京南瑞智慧交通科技有限公司 江苏南京 210031

每个神经元与上一层的所有神经元连接，每个连接都被赋予不同的权重值。

以输出层的第一个神经元为例，a_1 至 a_n 代表来自前一层神经元的全部数据，连接上的权重为 W_1 至 W_n。数据 a 与权重 W 相乘后加上偏置 b，经过激活函数 s 激活运算后，后层神经元输出结果为：

$$f(x)=\sigma(\sum_{i=1}^{n}W_i a_i+b) \tag{1}$$

图 1 右侧虚线框内描述的是自动微分的过程，可计算微分方程内的导数以及各项损失。

一般形式的微分方程表示为：

$$\begin{cases}\dfrac{\partial^k \boldsymbol{u}}{\partial t^k}(\boldsymbol{x},t)=D[\boldsymbol{u}(\boldsymbol{x},t)]\\ B[\boldsymbol{u}(\boldsymbol{x},t)]=h(\boldsymbol{x},t)\\ \dfrac{\partial^l \boldsymbol{u}}{\partial t^l}(\boldsymbol{x},0)=g(\boldsymbol{x})\end{cases} \tag{2}$$

使用深度学习网络求解式（2）的微分方程时，模型训练必不可少的是损失函数，网络在最小化损失函数的过程，不断地更新模型的参数，从而使其逼近微分方程的解。PINN 神经网络的损失库函数组成包括 3 类损失，即方程残差项、边界条件项、初始条件项 [2]。

方程残差项 MSE_u 是 PINN 的核心损失项，一般用于衡量神经网络预测的解是否满足微分方程。MSE_u 通过计算方程在一组随机采样点上的残差来实现，该损失项确保模型在训练过程中遵循物理定律，使得即使没有测量数据，也能通过物理约束来指导模型的学习。由式（2）的第一部分，可得方程残差项：

$$\mathrm{MSE}_u=\frac{1}{N_u}\sum_{i=1}^{N_u}\left|\frac{\partial^k \boldsymbol{u}}{\partial t^k}(\boldsymbol{x},t)-D[\boldsymbol{u}(\boldsymbol{x},t)]\right|^2 \tag{3}$$

边界条件项 MSE_b 用于衡量模型预测的解是否满足边界条件，这能确保模型在训练过程中从正确的起点开始，从而提高预测的准确性。由式（2）的第二部分，可得边界条件项：

$$\mathrm{MSE}_b=\frac{1}{N_b}\sum_{i=1}^{N_b}\left|B[\boldsymbol{u}(\boldsymbol{x},t)]-h(\boldsymbol{x},t)\right|^2 \tag{4}$$

初始条件项 MSE_I 用于衡量模型预测的解是否满足初始条件，确保模型在边界处的行为符合物理实际，避免在边界处出现不合理的偏差。由式（2）的第三部分，可得边界条件项：

$$\mathrm{MSE}_I=\frac{1}{N_I}\sum_{i=1}^{N_I}\left|\frac{\partial^l \boldsymbol{u}}{\partial t^l}(\boldsymbol{x},0)-g(\boldsymbol{x})\right|^2 \tag{5}$$

综合式（3）~（5），网络的总损失为：

$$\mathrm{Loss}=a\mathrm{MSE}_u+b\mathrm{MSE}_b+c\mathrm{MSE}_I \tag{6}$$

式中：a、b、c 为各损失项的权重系数，根据重要性平衡各项损失。通过最小化 Loss，使得模型的解趋近于解析解。

2 数值实验

为了探索 PINN 算法在求解非线性微分方程的优势以优化方案，本节将针对多种算例，利用 PINN 算法求解非线性微分方程，并研究不同激活函数、神经网络参数设置对训练效果的影响 [6]。

本文使用 Python 环境开展本工作中的所有计算，软件环境为 Python 3.8、PyTorch1.11、NumPy1.23.4、matplotlib 3.6.2，CUDA 版本为 11.3，cuDNN 版本为 8.2.1。测试采用的平台是 Intel(R) Core(TM) i7-13700KF（24 核，3.40 GHz）和 NVIDIA GeForce RTX 4090。

2.1 PINN 求解非线性微分方程

本节将以非线性常微分方程为例，基于 PINN 神经网络求解方法求解具体的非线性微分方程，分析求解效果。非线性常微分用公式表示为 [7]：

$$\begin{cases}y'''(x)-y^2(x)e^{-x}=0, x\in[0,1]\\ y(0)=1\\ y'(0)=1\\ y(1)=\mathrm{e}\end{cases} \tag{7}$$

该方程的解析解为：

$$y(x)=\mathrm{e}^x \tag{8}$$

神经网络选用 5 个隐藏层，每层 64 个神经元，在 [0，1] 之间随机选取 100 个数据点进行训练，使用 Adma 优化算法迭代至收敛。

图 2 为 PINN 结果与解析解的对比图，实线代表 PINN 的输出结果；虚线代表解析解的结果。观察曲线可以发现，两条曲线基本完全重合，PINN 的解与解析解基本一致，说明 PINN 求解偏微分方程取得了较好的效果，PINN 能够较好地求解非线性常微分方程。

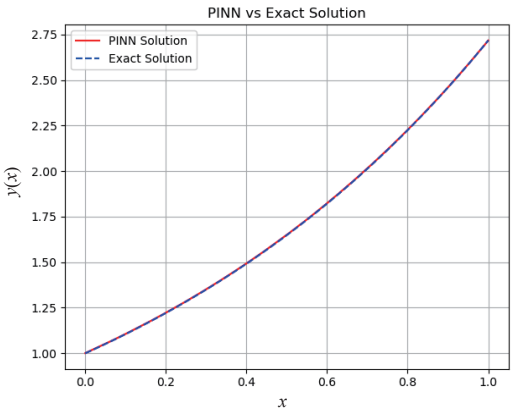

图 2 PINN 解与解析解对比

观察图 3 可以发现损失函数数值随迭代步数下降，在经过 41 次训练后，曲线下降速度减缓并趋向 0，由此可以判断模型已经收敛，最终 PINN 预测与解析解的误差低于 10^{-6} 并触发训练停止。综上所述，PINN 求解非线性微分方程时具有强大的拟合能力，模型容易收敛。

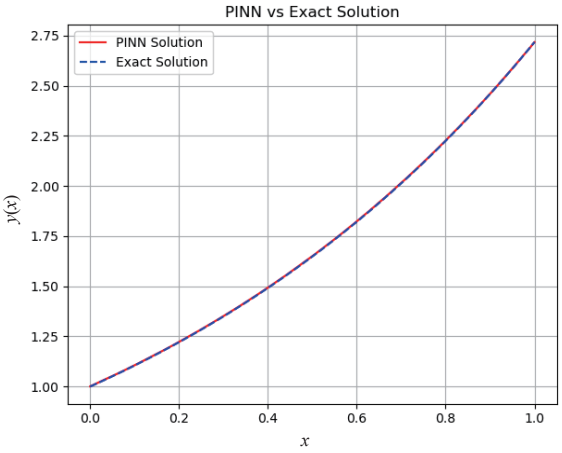

图 3 PINN 损失下降曲线

2.2 激活函数选取

激活函数是神经网络的关键组成部分，能够在神经元的输入中引入非线性变换，从而使网络具有学习和复杂表示的能力。为了探讨不同激活函数对微分方程求解的影响，并分析 PINN 任务中不同激活函数的性能差异，考虑非线性微分方程 [7]：

$$\begin{cases} y'''(x)+2\alpha\beta y(x)y'(x)+4\alpha^2 y'(x)=0, x\in[0,1] \\ y(0)=1 \\ y'(0)=0 \\ y(1)=0 \end{cases} \tag{9}$$

式中：α 取 0.0524；β 取 110。

常用的激活函数包括 tanh、ReLU、LeakyReLU、ELU 等，不同的激活函数在复杂度、缓解梯度消失等方面各具优势。

本节实验中网络层数固定为 5 层，每层神经元个数为 64，学习率为 0.000 1，训练次数为 10 000 epoch，若 1 000 个 epoch 训练无提升则触发早停。

激活函数训练结果对比如图 4 所示，图中横轴为输入值，竖轴代表预测值。由此可知，ReLU、ELU、LeakyReLU 训练的模型的预测值与实际值差距较大，而 tanh 和 Sigmoid 作为激活函数能够训练出更好的模型。激活函数 tanh 不仅引入非线性特性，在中间输入范围内 tanh 对输入变化敏感，能够更好地捕捉特征差异。激活函数 Sigmoid 虽然非线性特性较弱，但其在函数上平滑的特性亦有助于稳定地训练。

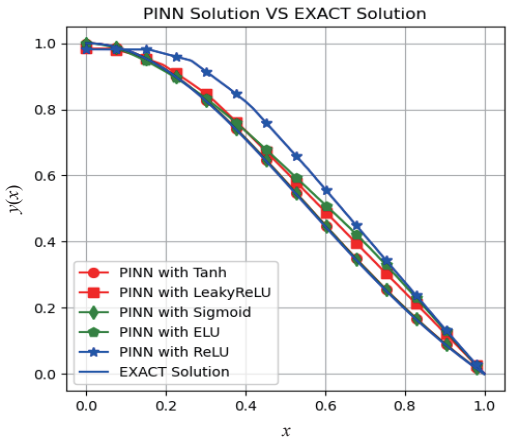

图 4 不同激活函数下 PINN 解对比

图 5 为不同激活函数下模型训练时损失函数数值变化的曲线，其中横轴代表迭代训练步数，竖轴是损失函数的值。由图 5 可知，ELU、RELU、LeakyRELU 作为激活函数时损失下降较慢且由于损失值在 1 000 步内没有提升而进入了早停，最终训练结果趋于欠拟合。tanh 和 Sigmoid 作为激活函数训练模型时损失函数收敛更快，并且收敛时 tanh 模型的损失值要低于 Sigmoid 模型，因此 tanh 更适合本节任务。

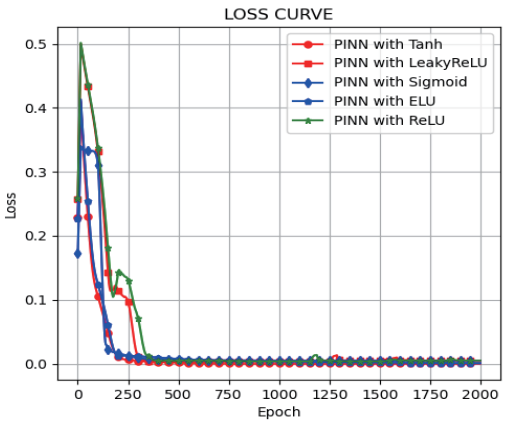

图 5 不同激活函数下 PINN 损失下降曲线

2.3 模型参数设置

PINN 网络深度通常为 4~10 层，每层 20~200 个神经元，具体参数设置取决于问题复杂度。此外，学习率的设置也影响着模型的性能。本节以式（10）为例 [7]，探讨不同网络参数对求解的影响。

$$\begin{cases} y'''(x)+y(x)y''(x)-(y'(x))^2+2=0, x\in[0,1] \\ y(0)=0 \\ y'(0)=0 \\ y(1)=0 \end{cases} \tag{10}$$

网络尺寸按照表格所示设置了 4 种，优化器选择 Adam，学习率设置为 0.000 1，早停迭代步数设置为 1 000，训练结果如表 1 所示。

表 1 多种网络尺寸下模型训练结果

网络尺寸	迭代步数	误差值 /RMSE
[5, 20]	10 000	4.508 811e-05
[5, 64]	3 233	9.861 776e-07
[8, 20]	10 000	9.872 882e-05
[8, 64]	4 566	9.900 445e-07

由表 1 可知，PINN 网络设置为 5 层且每层 20 神经元时网络尺寸较小，网络拟合能力有限，因此即便迭代次数上升，[5, 20] 网络损失依旧大于 [5, 64] 网络的损失值，属于欠拟合。类似的，[8, 20] 网络也属于欠拟合。对比 [5, 64] 网络和 [8, 64] 网络，网络深度加深但是性能并未进一步上升，损失值处于同一量级且相近，从迭代步数上看 [5, 64] 网络更节约时间和硬件资源。

网络尺寸按照前文的经验取 [5, 64]，优化器选择 Adam，学习率按照表格设置了 3 种，早停迭代步数设置为 1 000，训练结果如表 2 所示。

表 2 多种学习率下模型训练结果

学习率	迭代次数	误差值 /RMSE
0.01	2 300	3.746 522e-05
0.001	2 828	9.949 236e-07
0.000 1	3 233	9.861 776e-07
0.000 01	10 000	1.022 672e-05

表 2 中数据显示学习率选取为 0.001 和 0.000 1 时，相较于另外两种都达到了较好的结果。学习率选取为 0.01 时明显损失较大且由于损失值许久无改善而早停，这是因为学习率过大导致网络权重变化过大，难以达到最优解。选取为 0.000 01 时损失持续下降，但是学习率过小导致网络权重变化过小，迭代速度也因此变慢。由此可见，学习率选取为 0.001 和 0.000 1 时训练效果较好。

3 结论

本文聚焦 PINN 在微分方程中应用，通过数值实验研究 PINN 在非线性微分方程求解的效果。首先通过数值实验验证 5 层 64 神经元网络结构在 Adam 优化器作用下可实现与解析解小于 10^{-6} 的误差收敛；进一步针对非线性方程的激活函数敏感性测试显示，tanh 函数在导数传播中表现出更优的梯度保持能力。模型参数优化实验发现，网络深度超过 5 层后性能提升趋于饱和，学习率设置在 [0.0001,0.001] 区间可平衡收敛速度与稳定性。PINN 框架在非线性微分方程求解中展现出良好的泛化能力，通过合理设计网络结构与优化策略，可有效突破传统数值方法的局限性。

参考文献：

[1] 姚泽远，卜原玲，韩伟．基于深度神经网络与 Box-Behnken 响应面法的灵芝总三萜提取工艺优化 [J]. 南京工业大学学报 (自然科学版), 2023, 45(3): 347-354.

[2] 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]. Journal of computational physics, 2019, 378: 686-707.

[3] LU L, MENG X H, MAO Z P, et al. DeepXDE: a deep learning library for solving differential equations[J]. SIAM review, 2021, 63(1): 208-228.

[4] WANG Y F, ZHONG L L. NAS-PINN: neural architecture search-guided physics-informed neural network for solving PDEs[J]. Journal of computational physics, 2024, 496: 112603.

[5] LI X F, WANG H Y, YANG Y, et al. Deep learning-based solution for the KdV-family governing equations of ocean internal waves[J]. Ocean modelling, 2025, 194: 102493.

[6] 陈新海，刘杰，万仟，等．一种改进的基于深度神经网络的偏微分方程求解方法 [J]. 计算机工程与科学，2022, 44(11): 1932-1940.

[7] WANG W. Discussion on solving differential equations by finite difference method and PINN method[J]. Advances in applied mathematics, 2023, 12(7): 3298-3310.

【作者简介】

葛淼彦（1996—），男，江苏扬州人，硕士研究生，助理实验员，研究方向：机器学习、计算机视觉。

李凯锋（1995—），男，河南驻马店人，硕士研究生，助理工程师，研究方向：信息化集成、工业数字化平台体系建设、机器学习。

（收稿日期：2025-05-08 修回日期：2025-09-22）