Advances in Applied Mathematics 应用数学进展, 2026, 15(5), 484-499
Published Online May 2026 in Hans. https://www.hanspub.org/journal/aam
https://doi.org/10.12677/aam.2026.155245

Hans汉斯

自适应损失平衡物理信息神经网络算法求解几类偏微分方程的数值解

胡家强，白通拉嘎*

内蒙古工业大学理学院，内蒙古 呼和浩特

收稿日期：2026年4月21日；录用日期：2026年5月15日；发布日期：2026年5月26日

摘　要

物理信息神经网络(PINN)作为将物理定律与深度学习相结合的新型计算方法，受到广泛关注。物理信息神经网络的应用已初显成效，但如何提高其有限的精度仍是PINN面临的重大挑战。自适应损失平衡物理信息神经网络(lbPINN)通过建立高斯概率模型，为每个损失项设置自适应权重来定义自适应损失函数，解决了传统物理信息神经网络在求解偏微分方程时面临的损失函数权重分配难题。本研究通过应用PINN和lbPINN两种算法系统求解一维热传导方程、二维泊松方程、一维线性对流方程及一维电报方程，数值实验表明lbPINN相比于经典的PINN表现出更优的性能，相对L2误差有不同程度的降低，证实了lbPINN在求解这几类偏微分方程中表现出更高的精确度和有效性。

关键词

物理信息神经网络，自适应权重，自适应损失平衡物理信息神经网络

Self-Adaptive Loss Balanced Physics Informed Neural Networks Algorithm for Solving Numerical Solutions of Several Classes of Partial Differential Equations

Jiaqiang Hu, Tonglaga Bai*

College of Science, Inner Mongolia University of Technology, Hohhot Inner Mongolia

Received: April 21, 2026; accepted: May 15, 2026; published: May 26, 2026

*通讯作者。

文章引用: 胡家强, 白通拉嘎. 自适应损失平衡物理信息神经网络算法求解几类偏微分方程的数值解[J]. 应用数学进展, 2026, 15(5): 484-499. DOI: 10.12677/aam.2026.155245

Abstract

Physics-Informed Neural Networks (PINN), as a novel computational method integrating physical laws with deep learning, have garnered widespread attention. While PINN have demonstrated initial successes in applications, improving their limited accuracy remains a significant challenge. The Self-adaptive Loss Balanced Physics-Informed Neural Networks (lbPINN) addresses the difficulty of loss function weight allocation in solving partial differential equations faced by traditional PINN by establishing a Gaussian probability model and assigning adaptive weights to each loss term, thereby defining an adaptive loss function. In this study, both PINN and lbPINN algorithms were systematically applied to solve the one-dimensional heat equation, and one-dimensional telegraph equation. Numerical experiments show that, compared to the classical PINN, lbPINN exhibits superior performance, with varying degrees of reduction in relative L2 errors. This confirms that lbPINN demonstrates higher accuracy and effectiveness in solving these classes of partial differential equations.

Keywords

Physics-Informed Neural Networks, Adaptive Weighting, Self-Adaptive Loss Balanced Physics-Informed Neural Networks

1. 引言

偏微分方程(Partial Differential Equation, PDEs)是数学中极其重要的一种方程，其涉及未知多变量函数以及关于这些变量的偏导数。偏微分方程用来描述物理世界中的连续变化现象，如热传导[1]、声波传播[2]、电磁场变化[3]等，但 PDE 本身的性质，使得很难求出其精确解，甚至没有精确解。因此，PDEs 的数值求解就凸显得十分重要，在传统数值算法求解如有限差分方法[4] [5] (Finite Difference Method, FDM)和有限元方法[6] [7] (Finite Element Method, FEM)中，通常需要我们建立网格来辅助计算差分。

近年来，随着科学机器学习[8]以及自动微分技术[9] [10]的不断发展，人工神经网络等深度学习技术已在求解 PDEs 中得到了广泛的研究和应用。神经网络求解 PDEs 的基本原理主要是利用其强大的函数逼近能力[11]，将 PDEs 的求解转化为一个优化问题。近年来，物理信息神经网络[12] (Physics-Informed Neural Networks, PINN)的成功应用标志着科学计算领域内求解偏微分方程的一种新方法，在求解 PDEs 的效率性和准确性引起了广泛关注。PINN 的核心思路是在神经网络框架中将数学物理方程嵌入，利用损失函数的最小化过程，不断迭代更新神经网络的权重和偏置，使得通过神经网络得到的求解结果更加接近真实的物理现象，而且其损失函数通过软约束将 PDEs 作为唯一解，并非依赖大数据来确定关系，PINN 能够在更短的时间内构建网络来解决 PDEs 问题。然而，如何提高其有限的精度仍是 PINN 面临的重大挑战。因此，为了弥补这一差距，PINN 的改进版本不断涌现[13]-[15]。

为了更好地提高 PINN 算法的性能，对于激活函数和采样方法出现了各种类型的改进方法，例如特殊激活函数[16] [17]、残差点的自适应采样[18]等。自适应损失平衡物理信息神经网络[19] (Self-adaptive Loss Balanced Physics-Informed Neural Networks, lbPINN)在普通物理信息神经网络的基础上，通过建立高斯概率模型，基于多元高斯分布的极大似然估计设计了权重自适应调节算法，旨在自动调整损失函数中

各部分损失项的比例，使用自适应权重参数集合来确定损失的不确定性。在训练过程的每次迭代中自动更新每个损失项的权重，降低手动调节权重的难度。

本文我们将 PINN 算法和 lbPINN 算法应用到求解几类偏微分方程，并对比两种算法的性能。第二节将回顾 PINN 算法和 lbPINN 算法的主要思想以及自动微分算法，给出两种算法求解 PDEs 的参数限定和运行步骤；第三节展开数值实验，应用两种算法分别对一维热传导方程、二维泊松(Poisson)方程、一维线性对流方程及一维电报方程进行求解，对比两种算法在数值精度、物理约束满足的综合性能，验证对比两种算法的有效性和稳定性；第四节总结算法优缺点，并提出 lbPINN 算法潜在的应用前景。

2. 算法基本原理

2.1. 物理信息神经网络(PINN)

物理信息神经网络是一类深度学习模型，用于解决科学计算和工程问题中的偏微分方程。PINN 核心思想是利用神经网络的强大函数逼近能力来表示 PDEs 的解，将描述物理系统的控制方程作为先验知识，以软约束的形式嵌入到深度神经网络的训练过程中，无需或仅需少量标注数据即可实现高精度的解估计。本节详细阐述用于求解偏微分方程的物理信息神经网络框架。

考虑一个一般形式的偏微分方程，定义在域 $\Omega\in\mathbb{R}^d$ 上，时间区间为 $[0,T]$：

$$N\left[x,t;u(x,t)\right]=f(x,t),(x,t)\in\Omega\times[0,T] \tag{1}$$

$$B\left[x,t;u(x,t)\right]=g(x,t),(x,t)\in\partial\Omega\times[0,T] \tag{2}$$

$$I\left[x,0;u(x,0)\right]=h(x),x\in\Omega \tag{3}$$

其中：N 是一个非线性微分算子；$u(x,t)$ 是我们待求的未知解函数；B 和 I 分别表示边界条件和初始条件对应的算子；$f(x,t)$ 为方程非齐次项，$g(x,t)$ 为边界条件，$h(x)$ 为初始条件，且均为已知函数。

2.2. 网络结构与自动微分

我们构建一个多层前馈神经网络来逼近目标解，我们考虑深度为 L 的前馈全连接神经网络，输入为时间和空间坐标 (x,t)，用 $\hat{u}(x,t;\theta)$ 表示输入 $u(x,t)$ 的神经网络输出解，其中 θ 表示一组网络参数。一般前馈多层神经网络由一个输入层，$l-1$ 个隐藏层以及一个输出层组成，神经网络定义为：

$$\begin{cases}\hat{u}^{[l]}=x\\ \hat{u}^{[l]}=\sigma\left(W^{[l]}\hat{u}^{[l-1]}+b^{[l]}\right),l=2,3,\cdots,L-1\\ \hat{u}^{[l]}=\left(W^{[L]}\hat{u}^{[L-1]}+b^{[l]}\right)\end{cases} \tag{4}$$

式中，σ 为激活函数[20] [21]，通常使用 Tanh、Sigmoid、Softplus 和 Relu 等。$W^{[l]}$ 和 $b^{[l]}$ 分别表示第 l 层的权重和偏置。所有的权重矩阵和偏置向量都可以用一个参数集合 $\theta=\left\{W^{[l]},b^{[l]}\right\}_{1\le l\le L}$ 来表示。

自动微分技术是一种介于符号微分和数值微分之间的算法，该计算方法能够精确地计算出可微函数在某一点的导数值，在反向传播算法过程中得到了广泛应用。自动微分的核心问题是如何计算复杂函数的导数，通常是指多层复合函数在某一点的导数、梯度，具有极强的计算灵活性，可以完全向使用者隐藏求导的复杂过程。由于其仅仅是在基本的函数或常量中运用了符号微分法则，因此可以灵活地与编程语言的循环、分支等结构结合起来。根据链式法则，借助计算图计算出任意复杂函数的导数。

2.3. 损失函数设计

PINN 将物理先验知识整合到损失函数中，以增强数据的信息含量。PINN 的训练通过最小化一个精心设计的损失函数来实现，该函数将物理规律、边界条件和初始条件统一在一个优化框架内，通常由三个损失项组成：

$$L_{\text{total}}\left(\theta;N\right)=\lambda_f L_{PDE}\left(\theta;N_f\right)+\lambda_b L_{BC}\left(\theta;N_b\right)+\lambda_i L_{IC}\left(\theta;N_i\right) \tag{5}$$

式(5)中，$\lambda=\left\{\lambda_f,\lambda_b,\lambda_i\right\}$ 为损失函数的权重参数，用于平衡各项损失项的重要性。L_{PDE} 表示在计算区域内部样本点 $\left\{x_{PDE}^j,t_{PDE}^j\right\}_{j=1}^{N_f}$ 的偏微分方程的损失，该项强制网络近似解在计算域内部满足控制方程；L_{BC} 表示在计算区域边界训练点 $\left\{x_{BC}^j,t_{BC}^j\right\}_{j=1}^{N_b}$ 的边界条件的损失，该项确保解在边界上满足给定条件；L_{IC} 表示在计算区域初始训练点 $\left\{x_{IC}^j,t_{IC}^j\right\}_{j=1}^{N_i}$ 的初始条件的损失，该项确保解在边界上满足给定条件。N_f、N_b、N_i 分别是计算区域内部训练点、边界训练点和初始训练点的个数。L_{PDE}、L_{BC}、L_{IC} 定义如下：

$$\begin{cases} L_{PDE}\left(\theta;N_f\right)=\dfrac{1}{N_f}\displaystyle\sum_{i=1}^{N_f}\left|N\left(x_{PDE}^i,t_{PDE}^i;\theta\right)-f\left(x_{PDE}^i,t_{PDE}^i\right)\right|^2 \\ L_{BC}\left(\theta;N_b\right)=\dfrac{1}{N_b}\displaystyle\sum_{i=1}^{N_b}\left|B\left(x_{BC}^i,t_{BC}^i;\theta\right)-g\left(x_{BC}^i-t_{BC}^i\right)\right|^2 \\ L_{IC}\left(\theta;N_i\right)=\dfrac{1}{N_i}\displaystyle\sum_{i=1}^{N_i}\left|I\left(x_{IC}^i,t_{IC}^i;\theta\right)-h\left(x_{IC}^i\right)\right|^2 \end{cases} \tag{6}$$

2.4. 训练过程

网络的训练过程转化为一个优化问题：

$$\theta^*=\arg\min_{\theta}L_{\text{total}}\left(\theta;N\right) \tag{7}$$

我们通常使用基于梯度的优化算法[22]-[24] (例如 SGD、Adam、L-BFGS 等)来迭代更新网络参数 θ。在每一步迭代中，计算损失函数关于参数的梯度，并据此更新参数，使得网络预测的解同时满足 PDE 本身、边界条件和初始条件。训练完成后，网络 $\hat{u}\left(x,t;\theta^*\right)$ 即可作为偏微分方程的一个连续、可微的近似解。下面算法 1 总结了 PINN 算法的参数限定和运行步骤：

算法一 物理信息神经网络(PINN)

参数设定：训练步数为 S，学习率为 l_r

目标：找出参数的最佳模型

开始

步骤 1：考虑一个具有初始参数 θ 的物理信息神经网络 $\hat{u}\left(x,t;\theta\right)$

步骤 2：指定训练点 $N=\left\{N_f,N_b,N_i\right\}$

步骤 3：使用梯度下降算法经过 S 步更新参数 θ

循环 从 $s=1$ 到 $s=S$

(a) 根据式(6)定义损失函数式(5) $L_{\text{total}}\left(\theta;N\right)$

(b) 通过 Adam 更新参数 θ：

$$\theta_{s+1}\leftarrow\text{Adam}\left(L_{\text{total}}\left(\theta_s;N\right),l_r\right)$$

结束

2.5. 自适应损失平衡物理信息神经网络(lbPINN)

自适应损失平衡方法的主要思想来源于一篇基于不确定性高斯似然最大化来权衡多任务深度学习中多个损失函数的论文[25]。基于“多任务深度学习”和“不确定性”的思想，项子学等人[19]提出了自适应损失平衡物理信息神经网络(lbPINN)。自适应损失平衡方法的主要思想是基于最大化多任务深度学习问题的不确定性的高斯似然来对多个损失函数进行加权。在普通物理信息神经网络的基础上，加入自适应损失加权法，在每次训练迭代开始时动态调整单个损失项的相对重要性，确保在没有单个损失分量占主导地位的情况下实现平衡的优化。物理信息神经网络的自适应损失平衡算法如下：

首先建立一个输出为u的高斯概率模型，其中有 PINN 的输出近似值$\hat{u}(x,t;\theta)$和不确定性参数ε：

$$p\left(u\middle|\hat{u}(x,t;\theta)\right)=N\left(\hat{u}(x,t;\theta),\varepsilon^2\right) \tag{8}$$

考虑输出服从高斯分布，神经网络的不确定性参数通常是固定作为权重衰减的一部分，为了捕获依赖数据的任意不确定性，将根据最大似然推断来调整不确定性参数。基于最小化目标，模型的负对数似然为：

$$\begin{aligned}-\log p\left(u\middle|\hat{u}(x,t;\theta)\right)&\propto\frac{1}{2\varepsilon^2}\left\|u-\hat{u}(x,t;\theta)\right\|^2+\log\varepsilon\\&=\frac{1}{2\varepsilon^2}L_1(\theta)+\log\varepsilon\end{aligned} \tag{9}$$

式(9)中，L_1表示输出变量的损失。我们将式(6)中的损失项代入，则可以得到：

$$\begin{aligned}L(\varepsilon;\theta;N)=&\frac{1}{2\varepsilon_f^2}L_{PDE}(\theta;N_f)+\frac{1}{2\varepsilon_b^2}L_{BC}(\theta;N_b)\\&+\frac{1}{2\varepsilon_i^2}L_{IC}(\theta;N_i)+\log\varepsilon_f\varepsilon_b\varepsilon_i\end{aligned} \tag{10}$$

式(10)中，$\varepsilon=\{\varepsilon_f,\varepsilon_b,\varepsilon_i\}$是可训练的参数，$\frac{1}{2\varepsilon^2}$表示损失项的自适应权重，用于平衡不同损失项的重要性，称这种具有自适应损失平衡方法的物理信息神经网络为 lbPINN。在训练过程中，lbPINN 同时优化神经网络参数θ和自适应权重参数ε。

当一个损失项的ε值增加时，其对应的损失权重会减小,这意味着该项的不确定性较高，因此在训练过程中其重要性会降低。相反，损失权重的增加意味着该项的不确定性较低，其重要性应该增加。以边界损失L_{BC}对应的不确定参数ε_b为例，对L关于ε_b求偏导：

$$\frac{\partial L}{\partial\varepsilon_b}=\frac{\partial}{\partial\varepsilon_b}\left(\frac{1}{2\varepsilon_b^2}L_{BC}+\log\varepsilon_b\right)=-\frac{L_{BC}}{\varepsilon_b^3}+\frac{1}{\varepsilon_b}=\frac{1}{\varepsilon_b}\left(1-\frac{L_{BC}}{\varepsilon_b^2}\right)$$

我们通过$\frac{L_{BC}}{\varepsilon_b^2}$的大小，分三种情况说明$\varepsilon_b$和权重的动态变化：

情况一：若训练初期L_{BC}很大(远大于ε_b^2)，此时$\frac{L_{BC}}{\varepsilon_b^2}\gg1$，代入偏导数公式得$\frac{\partial L}{\partial\varepsilon_b}<0$。当梯度为负时，在梯度下降过程中$\varepsilon_b$会增大，而权重是$\frac{1}{2\varepsilon_b^2}$，$\varepsilon_b$增大将会使得权重减小，边界损失的重要性降低；

情况二：若训练中后期L_{BC}该项损失逐渐收敛，即L_{BC}变小，此时$\frac{L_{BC}}{\varepsilon_b^2}\approx1$，有$\frac{\partial L}{\partial\varepsilon_b}\approx0$，$\varepsilon_b$不再剧烈增长，甚至可能略微减小，这意味着边界损失的不确定性降低，权重趋于稳定，重要性回归合理水平；

情况三：若 $L_{BC}<\varepsilon_b^2$，此时 $\frac{L_{BC}}{\varepsilon_b^2}<1$，代入偏导数公式得 $\frac{\partial L}{\partial \varepsilon_b}>0$，当梯度为正时，在梯度下降过程中 ε_b 会减小，使得权重变大，边界损失的重要性升高。

这里，我们也可以使用对数方差 $s=\{s_f,s_b,s_i\}$ (其中 $s=\log\varepsilon^2$)来表示，则自适应损失平衡物理信息神经网络的损失函数可表示为：

$$\begin{aligned}L(s;\theta;N)=&\frac{1}{2}\mathrm{e}^{-s_f}L_{PDE}(\theta;N_f)+\frac{1}{2}\mathrm{e}^{-s_b}L_{BC}(\theta;N_b)\\&+\frac{1}{2}\mathrm{e}^{-s_i}L_{IC}(\theta;N_i)+\frac{1}{2}(s_f+s_b+s_i)\end{aligned}\tag{11}$$

我们的目标是通过使用基于梯度的优化器[22]-[24]最小化损失 $L(s;\theta;N)$ 来找到最佳模型权重值 θ^* 和合适的自适应权重 s^*。下面算法 2 总结了 lbPINN 算法的参数限定和运行步骤：

算法二 自适应损失平衡物理信息神经网络(lbPINN)

参数设定：训练步数为 S，学习率为 l_r，损失函数权重的初始值为 $\varepsilon=\{\varepsilon_f,\varepsilon_b,\varepsilon_i\}$

目标：找到参数为 θ^* 和权重 ε^* 的最优模型

开始

步骤 1：设置一个具有初始参数 θ 的物理信息神经网络 $\hat{u}(x,t;\theta)$

步骤 2：指定训练点 $N=\{N_f,N_b,N_i\}$

步骤 3：构建一个高斯概率模型，其均值由 PINN 的输出和参数 ε 给出

步骤 4：使用梯度下降法经过 S 步将参数 θ 和 ε 更新为：

循环 从 $s=1$ 到 $s=S$

(a) 根据极大似然估计定义加权损失函数 $L(\varepsilon;\theta;N)$

(b) 通过 Adam 调整参数 ε，使满足约束的概率最大化：

$$\varepsilon_{s+1}\leftarrow \mathrm{Adam}(L(\varepsilon_s;\theta_s;N),l_r)$$

(c) 通过 Adam 更新参数 θ：

$$\theta_{s+1}\leftarrow \mathrm{Adam}(L(\varepsilon_s;\theta_s;N),l_r)$$

结束

3. 数值实验

3.1. 一维热传导方程

热传导方程是一个描述热量在物体内部或区域内的传播规律以及温度如何随时间变化的物理模型。该偏微分方程为：

$$u_t=au_{xx},x\in[0,1],t\in[0,1],a=0.04\tag{12}$$

边界条件：$u(0,t)=0$，$u(1,t)=0$

初始条件：$u(x,0)=50\sin(\pi x)$

精确解为：$u(x,t)=50\mathrm{e}^{-a\pi^2t}\sin(\pi x)$

实验中 PINN 和 lbPINN 均为前馈全连接神经网络，输入层有 2 个神经元，输出层有 1 个神经元，隐

藏层有 6 层，每层 40 个神经元，隐藏层使用双曲正切函数 tanh 为激活函数。神经网络使用 Adam 优化器以 0.001 为学习率迭代 7500 次，训练点设置为 $N_f = 600$ 、$N_i = 40$ 、$N_b = 100$，初始令 $\varepsilon = \{\varepsilon_f, \varepsilon_b, \varepsilon_i\}$ 全为 1，即 $s = \{s_f, s_b, s_i\}$ 为 0，编程语言采用 Python。本节实验判断一维热传导方程的误差定义为：

$$\text{L2 error} = \frac{\sqrt{\sum_{i=1}^{N} \left|\hat{u}(x_i,t_i) - u(x_i,t_i)\right|^2}}{\sqrt{\sum_{i=1}^{N} \left|u(x_i,t_i)\right|^2}}$$

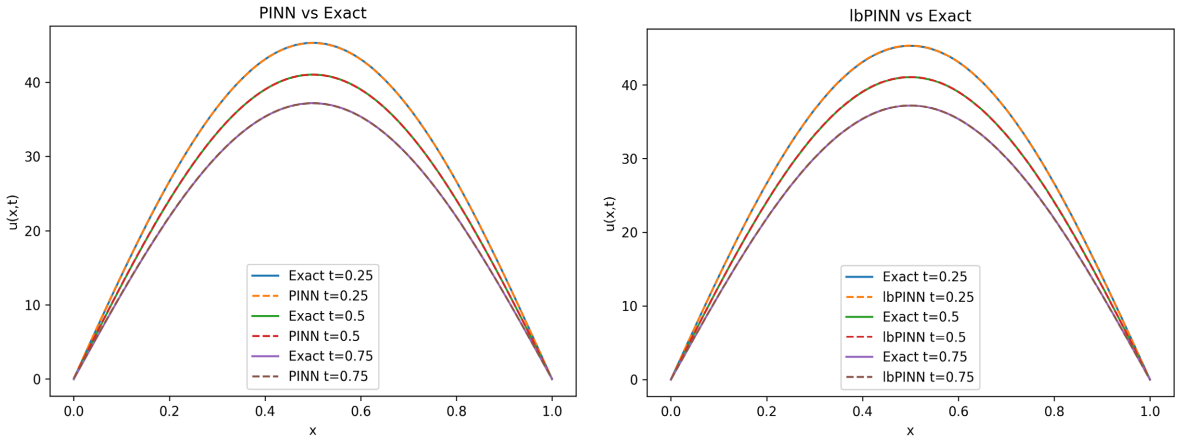
本实验的损失函数为：$L_{\text{total}}(\theta;N) = \lambda_f L_{PDE}(\theta;N_f) + \lambda_b L_{BC}(\theta;N_b) + \lambda_i L_{IC}(\theta;N_i)$

lbPINN 将自适应权重分别用于下面三个子损失：

偏微分方程的损失：$L_{PDE} = \frac{1}{N_f}\sum_{i=1}^{N_f} \left|u_t(x_i,t_i) - a u_{xx}(x_i,t_i)\right|^2$

初始条件的损失：$L_{IC} = \frac{1}{N_i}\sum_{i=1}^{N_i} \left|u(x_i,0) - \sin(\pi x_i)\right|^2$

边界条件的损失：$L_{BC} = \frac{1}{N_b}\sum_{i=1}^{N_b} \left[\left|u(0,t_i) - 0\right|^2 + \left|u(1,t) - 0\right|^2\right]$



(a) (b)

Figure 1. (a) Comparison of the numerical solution and the exact solution of the PINN algorithm. (b) Comparison of the numerical solution and the exact solution of the lbPINN algorithm

图 1. (a) PINN 算法数值解与精确解对比图；(b) lbPINN 算法数值解与精确解对比图

图 1 展示了一维热传导方程在 t = 0.25、t = 0.5、t = 0.75 三个时刻 PINN 算法与 lbPINN 算法的数值解与精确解的对比。总体而言，两种算法均能模拟热传导过程的基本特性，清晰展现温度随时间的变化。图 1(a)呈现了一维热传导方程的精确解与 PINN 算法数值解的对比图，图 1(b)展示了方程精确解与 lbPINN 算法数值解的对比图，通过对比发现，lbPINN 算法的数值解曲线与精确解曲线重合度高于传统的 PINN 算法。特别是在峰值区域，lbPINN 对解空间特征捕捉更为精确。这表明通过算法改进，lbPINN 在保持物理约束的同时，显著提升了方程解的逼近精度，为偏微分方程数值求解提供了更有效的深度学习方案。

经过深入对比，表 1 展示了 PINN 算法与 lbPINN 算法的数值解与精确解之间的相对 L2 误差。两组 L2 相对误差表明：lbPINN (0.101%)相比 PINN (0.095%)误差下降，说明自适应损失平衡能够有效地平衡

一维热传导方程训练中的多任务约束，从而使模型对幅值与相位的拟合更准确，表明 lbPINN 算法的性能显著提升，稳定性增强。

Table 1. L2 relative errors of the PINN and lbPINN algorithms for the one-dimensional heat conduction equation
表 1. 一维热传导方程 PINN 算法与 lbPINN 算法的 L2 相对误差

算法	L2 relative error
PINN	1.0108e−03
lbPINN	9.5124e−04

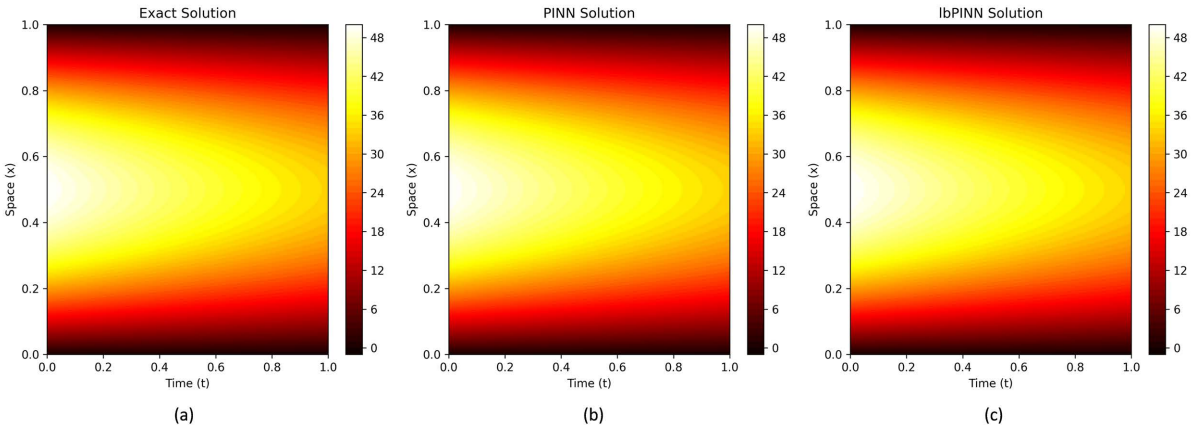


Figure 2. (a) Exact solution; (b) Numerical solution of Algorithm 1 (PINN); (c) Numerical solution of Algorithm 2 (lbPINN)
图 2. (a) 精确解；(b) 算法 1 (PINN)的数值解；(c) 算法 2 (lbPINN)的数值解

如图 2 中三幅热力图所示，我们对比了精确解、PINN 算法数值解以及 lbPINN 算法数值解。三幅图像在整体上呈现出高度相似的色彩分布模式，即温度场在空间维度上由高温区向低温区平滑过渡，在时间维度上展现出与物理规律一致的演化过程。值得注意的是，虽然 PINN 与 lbPINN 的解在整体形态上与精确解极为接近，但通过热力图色彩的细微差异仍可辨识，lbPINN 的解在高温区域(色彩梯度变化最显著处)与精确解的重合度更高。这表明，相较于 PINN 方法，lbPINN 在捕捉一维热传导方程的细节特征，尤其是在高梯度区域，具有更高的精度和更好的性能，从而更准确地复现了真实的物理场分布。

3.2. 二维泊松(Poisson)方程

二维泊松方程是数学物理中最核心的椭圆型偏微分方程之一，广泛应用于电磁学、流体力学等领域，理解它不仅是掌握偏微分方程的基础，也是解决诸多工程与物理问题的关键。该偏微分方程为：

$$-\Delta u(x,y)=f(x,y),(x,y)\in[0,1]^2,u|_{\partial\Omega}=u^*(x,y) \tag{13}$$

取制造解为：$u^*(x,y)=\sin(\pi x)\sin(\pi y)$。

则有：$\Delta u^*(x,y)=-2\pi^2\sin(\pi x)\sin(\pi y)$，$f(x,y)=-\Delta u^*(x,y)=2\pi^2\sin(\pi x)\sin(\pi y)$。

实验中的 PINN 和 lbPINN 均为前馈全连接神经网络，输入层有 2 个神经元，输出层有 1 个神经元，隐藏层有 6 层，每层 80 个神经元，隐藏层间以双曲正切函数为激活函数。神经网络使用 Adam 优化器以 0.001 为学习率迭代 8000 次，训练点设置为 $N_f=4000$、$N_b=800$，初始令 $\varepsilon=\{\varepsilon_f,\varepsilon_b\}$ 全为 1，即 $s=\{s_f,s_b\}$ 全为 0，编程语言采用 Python。本节实验判断二维泊松方程的误差定义为：

$$\text{L2 error} = \frac{\sqrt{\sum_{i=1}^{N}\left|\hat{u}(x_i,t_i)-u^*(x_i,t_i)\right|^2}}{\sqrt{\sum_{i=1}^{N}\left|u^*(x_i,t_i)\right|^2}}$$

本实验的损失函数为：$L_{\text{total}}(\theta;N)=\lambda_f L_{PDE}(\theta;N_f)+\lambda_b L_{BC}(\theta;N_b)$。

lbPINN 将自适应权重分别用于下面两个子损失：

偏微分方程的损失：$L_{PDE}=\frac{1}{N_f}\sum_{i=1}^{N_f}\left|-\left(u_{xx}(x_i,y_i)+u_{yy}(x_i,y_i)\right)-f(x_i,y_i)\right|^2$。

边界条件的损失：$L_{BC}=\frac{1}{N_b}\sum_{i=1}^{N_b}\left|u(x_i,y_i)\right|_{\partial\Omega}-u^*(x_i,y_i)\right|^2$。

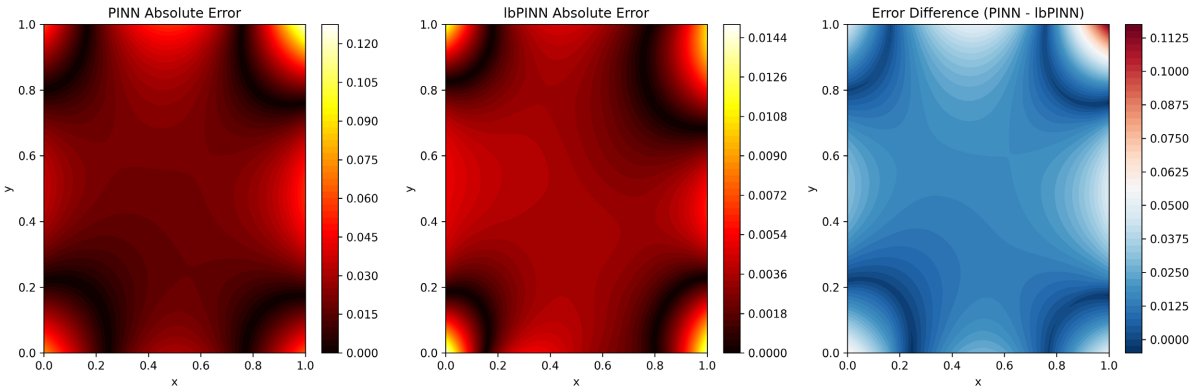


Figure 3. Comparison of absolute errors and error differences between PINN and lbPINN in solving the two-dimensional Poisson equation
图 3. 二维泊松方程求解中 PINN 与 lbPINN 的绝对误差及误差差异对比

图 3 给出了二维泊松方程求解中 PINN 与 lbPINN 的绝对误差及误差差异对比。通过对比二维泊松方程的绝对误差分布可见，lbPINN 算法相较于 PINN 算法在求解精度上具有显著优势。左侧 PINN 误差图中呈现大范围红色高误差区域，而中间 lbPINN 误差图整体呈现浅黄色调，最大误差显著降低。右侧误差差异图更清晰地展示了改进效果：深蓝色区域集中分布在方程求解域的中心及边界区域，表明这些关键区域的误差得到明显改善。这三组热力图共同验证了 lbPINN 算法在二维泊松方程求解中具有更优的数值稳定性和计算精度，特别是在梯度变化剧烈的区域表现出更强的适应性。

Table 2. L2 relative errors of the PINN and lbPINN algorithms for the two-dimensional Poisson equation
表 2. 二维泊松方程 PINN 算法与 lbPINN 算法的 L2 相对误差

算法	L2 relative error
PINN	4.7588e−02
lbPINN	6.7813e−03

表 2 展示了 PINN 算法与 lbPINN 算法求解二维泊松方程的 L2 相对误差。在同样的网络与采样设置下，PINN 算法的 L2 相对误差为 4.7588×10^{-2}，而引入自适应损失的 lbPINN 降至 6.7813×10^{-3}，相对下降约 85.7%，总体精度提升约 7 倍，说明 lbPINN 相比 PINN 精度更高，能得到更接近精确解的结果，性

能显著提升，实现了更平衡的优化路径与更稳的收敛。误差降低一个数量级的显著改进，验证了 lbPINN 算法在求解二维泊松方程时，对解空间的逼近能力显著增强，能够更精确地捕捉方程的物理特征。

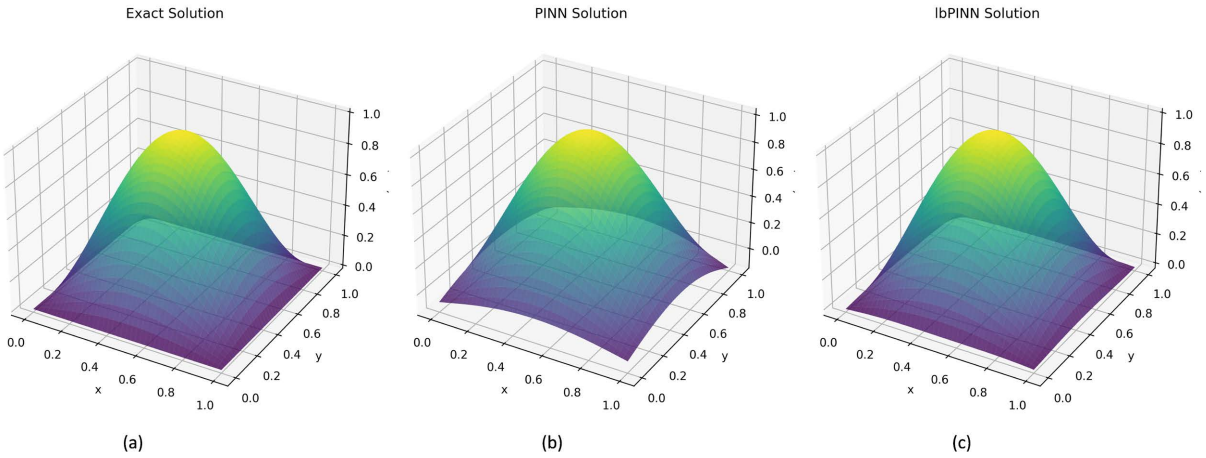


Figure 4. (a) Exact solution; (b) Numerical solution of Algorithm 1 (PINN); (c) Numerical solution of Algorithm 2 (lbPINN)
图 4. (a) 精确解；(b) 算法 1 (PINN)的数值解；(c) 算法 2 (lbPINN)的数值解

在图 4 中，基于对二维泊松方程数值解的对比分析，通过观察精确解、PINN 算法数值解与 lbPINN 算法数值解的三维曲面图可以看到：虽然 PINN 与 lbPINN 的解在整体形态和空间分布趋势上与精确解基本一致，均呈现中部高、四周低的曲面特征，且颜色从紫色(低值)向黄色(高值)平滑过渡，但 lbPINN 解在曲面细节上更接近精确解。具体而言，PINN 解在曲面中部高梯度区域与精确解存在肉眼可辨的差异，而 lbPINN 解则更好地复现了精确解的曲率变化和极值分布。充分表明 lbPINN 在求解二维泊松方程时，对解空间的逼近能力更强，有效提升了数值求解的准确性。

3.3. 一维线性对流方程

一维线性对流方程是流体力学、空气动力学等领域中描述物质或物理量随流体流动而传递的基础偏微分方程，是理解“对流”这一输运现象的核心模型。该偏微分方程为：

$$u_t + au_x = 0, x \in [0,1], t \in [0,1] \tag{14}$$

初始条件：$u(x,0) = \sin(2\pi x)$。

周期边界：$u(0,t) = u(1,t)$，$u_x(0,t) = u_x(1,t)$。

精确解为：$u(x,t) = \sin(2\pi(x-at))$。

式(14)中取 $a=1$，实验中的 PINN 和 lbPINN 均为前馈全连接神经网络，输入层有 2 个神经元，输出层有 1 个神经元，隐藏层有 6 层，每层 60 个神经元，隐藏层间以双曲正切函数为激活函数。神经网络使用 Adam 优化器以 0.001 为学习率迭代 9000 次，训练点设置为 $N_f = 2500$、$N_i = 256$、$N_b = 256$，初始令 $\varepsilon = \{\varepsilon_f, \varepsilon_b, \varepsilon_i\}$ 全为 1，即 $s = \{s_f, s_b, s_i\}$ 全为 0，编程语言采用 Python。本节实验判断一维线性对流方程的误差定义为：

$$\text{L2 error} = \frac{\sqrt{\sum_{i=1}^{N} \left|\hat{u}(x_i,t_i) - u(x_i,t_i)\right|^2}}{\sqrt{\sum_{i=1}^{N} \left|u(x_i,t_i)\right|^2}}$$

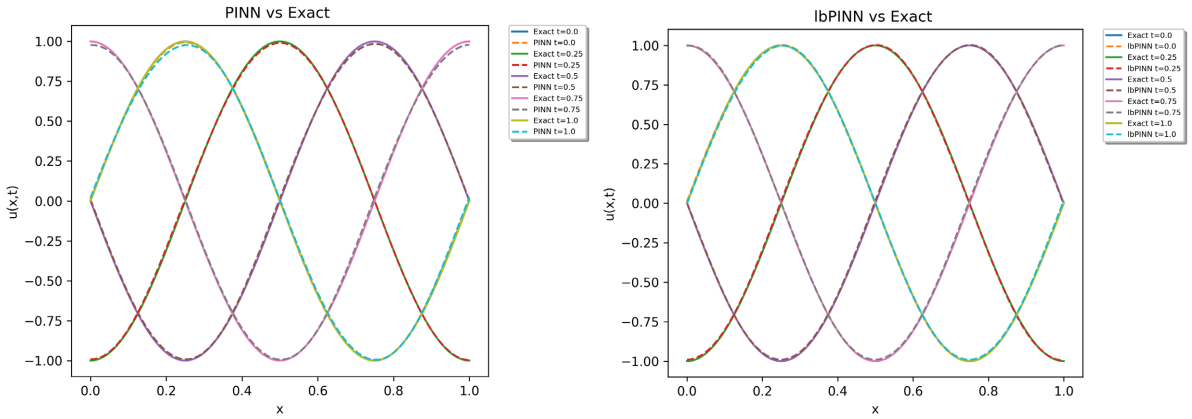
本实验损失函数为：$L_{\text{total}}(\theta;N)=\lambda_f L_{PDE}(\theta;N_f)+\lambda_b L_{BC}(\theta;N_b)+\lambda_i L_{IC}(\theta;N_i)$。

lbPINN 将自适应权重分别用于三个子损失：

偏微分方程的损失：$L_{PDE}=\frac{1}{N_f}\sum_{i=1}^{N_f}\left|u_t(x_i,t_i)-au_x(x_i,t_i)\right|^2$。

初始条件的损失：$L_{IC}=\frac{1}{N_i}\sum_{i=1}^{N_i}\left|u(x_i,0)-\sin(2\pi x_i)\right|^2$。

边界条件的损失：$L_{BC}=\frac{1}{N_b}\sum_{i=1}^{N_b}\left[\left|u(0,t_i)-u(1,t_i)\right|^2+\left|u_x(0,t_i)-u_x(1,t_i)\right|^2\right]$。



(a) (b)

Figure 5. (a) Comparison of the numerical solution and the exact solution of the PINN algorithm; (b) Comparison of the numerical solution and the exact solution of the lbPINN algorithm

图 5. (a) PINN 算法数值解与精确解对比图；(b) lbPINN 算法数值解与精确解对比图

图 5 分别展示了 PINN 和 lbPINN 在求解一维线性对流方程时，不同时刻数值解与精确解的对比情况。虽然两种算法均能大致捕捉到波动传播的整体趋势，但 lbPINN 算法在各时间下的预测曲线与精确解的重合度显著优于 PINN 方法。具体而言，在图 5(a)中，PINN 的数值解曲线与精确解曲线存在一定偏差，随着时间推移，这种偏差有逐渐显现和扩大的趋势。而在图 5(b)里，lbPINN 的数值解曲线与精确解曲线贴合得更为紧密，从初始时刻到最终时刻，各时间点的数值解都能较好地匹配精确解，体现出 lbPINN 在求解方程时，相比 PINN 具有更高的精度和更好的稳定性，能够更准确地捕捉对流过程中物理量的传播特征。

Table 3. L2 relative errors of the PINN and lbPINN algorithms for the one-dimensional linear convection equation

表 3. 一维线性对流方程 PINN 算法与 lbPINN 算法的 L2 相对误差

算法	L2 relative error
PINN	1.4223e−02
lbPINN	9.9976e−03

表 3 清晰地展示了 PINN 算法与 lbPINN 算法求解一维线性对流方程的 L2 相对误差。结果分析，改进型 lbPINN 算法相较于基础 PINN 算法在计算精度上取得显著提升。具体数据显示，PINN 算法的 L2 相对误差为1.4223×10^{-2}，而 lbPINN 算法将该误差降低至9.9976×10^{-3}，降幅达到约 29.7%。这一结果表明，

lbPINN 算法在求解一维线性对流方程时，能够更精确地描述波动传播特性，特别是在保持波形形态和相位准确性方面表现更优。误差的显著降低验证了 lbPINN 改进策略在双曲型偏微分方程求解中的有效性，为这类问题的数值计算提供了更可靠的解决方案。

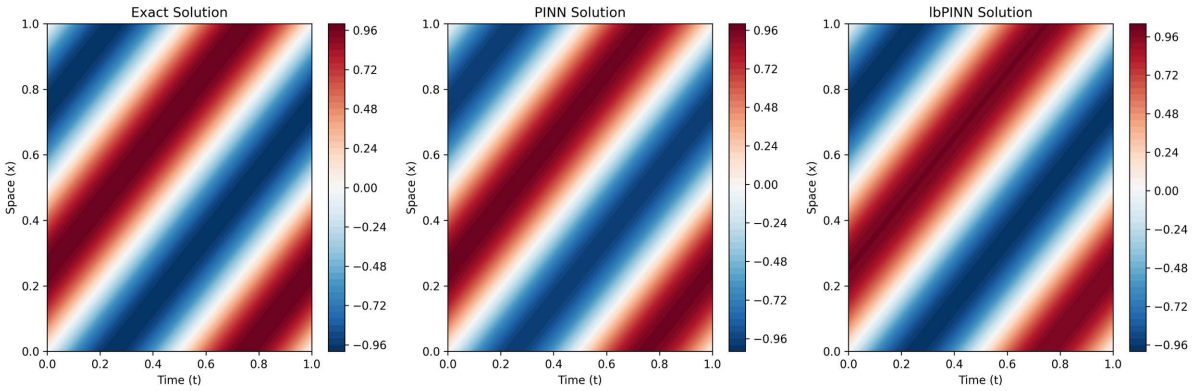


Figure 6. (a) Exact solution; (b) Numerical solution of Algorithm 1 (PINN); (c) Numerical solution of Algorithm 2 (lbPINN)
图 6. (a) 精确解；(b) 算法 1 (PINN)的数值解；(c) 算法 2 (lbPINN)的数值解

针对一维线性对流方程，通过图 6 对比精确解、PINN 算法数值解与 lbPINN 算法数值解的热力图可得出以下结论：三种解在整体上均呈现出由左下角($t=0, x=0$)向右上角($t=1, x=1$)传播的波动特征，且颜色从蓝色向红色呈现阶梯状过渡，符合线性对流方程的物理传播特性。然而在细节层面，PINN 解在传播路径上出现明显的数值耗散现象，表现为颜色过渡带存在横向扩散与模糊；而 lbPINN 解不仅保持了与精确解高度一致的尖锐过渡前沿，其传播轨迹的线性特征和颜色分层结构也更接近精确解。这一现象表明，lbPINN 显著提升了对一维线性对流方程中波动传播过程的建模精度，更好地保持了方程的物理特性。

3.4. 一维电报方程

一维电报方程用于描述传输线上电压或电流的传播，是研究波的传播与耗散等问题的基础模型之一。该偏微分方程为：

$$u_{tt}+\gamma u_t+\beta u=c^2u_{xx}, x\in[0,1], t\in[0,1] \tag{15}$$

初始条件：$u(x,0)=A\sin(k\pi x)$。

初始速度：$u_t(x,0)=-\frac{\gamma}{2}A\sin(k\pi x)$。

边界条件：$u(0,t)=u(1,t)=0$。

精确解为：$u(x,t)=\mathrm{e}^{-\frac{\gamma}{2}t}A\sin(k\pi x)\cos(\omega t), \omega=\sqrt{\beta+c^2k^2\pi^2-\left(\frac{\gamma}{2}\right)^2}$。

这里我们取 $\gamma=0.5, \beta=5.0, c=1.0, k=1, A=1$，实验中的 PINN 和 lbPINN 均为前馈全连接神经网络，输入层有 2 个神经元，输出层有 1 个神经元，隐藏层有 6 层，每层 60 个神经元，隐藏层间以双曲正切函数 tanh 为激活函数。神经网络使用 Adam 优化器以 0.001 为学习率迭代 9000 次，训练点设置为 $N_f=3000$、$N_{i1}=256$、$N_{i2}=256$、$N_b=256$，初始令 $\varepsilon=\{\varepsilon_f,\varepsilon_b,\varepsilon_{i1},\varepsilon_{i2}\}$ 全为 1，即 $s=\{s_f,s_b,s_{i1},s_{i2}\}$ 全为 0，编程语言采用 Python。本节实验判断一维电报方程的误差定义为：

$$\text{L2 error} = \frac{\sqrt{\sum_{i=1}^{N}\left|\hat{u}\left(x_i,t_i\right)-u\left(x_i,t_i\right)\right|^2}}{\sqrt{\sum_{i=1}^{N}\left|u\left(x_i,t_i\right)\right|^2}}$$

本实验损失函数为：

$$L_{\text{total}}\left(\theta;N\right)=\lambda_f L_{PDE}\left(\theta;N_f\right)+\lambda_b L_{BC}\left(\theta;N_b\right)+\lambda_{i1}L_{IC1}\left(\theta;N_{i1}\right)+\lambda_{i2}L_{IC2}\left(\theta;N_{i2}\right)$$

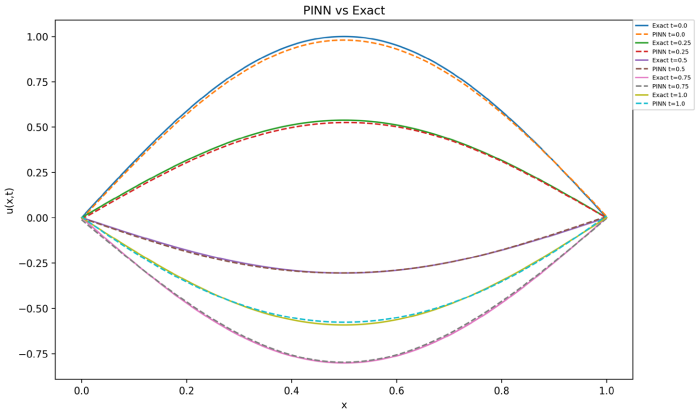
lbPINN 将自适应权重分别用于四个子损失：

偏微分方程的损失：$L_{PDE}=\frac{1}{N_f}\sum_{i=1}^{N_f}\left|u_{tt}\left(x_i,t_i\right)+\gamma u_t\left(x_i,t_i\right)+\beta u\left(x_i,t_i\right)-c^2u_{xx}\left(x_i,t_i\right)\right|^2$。

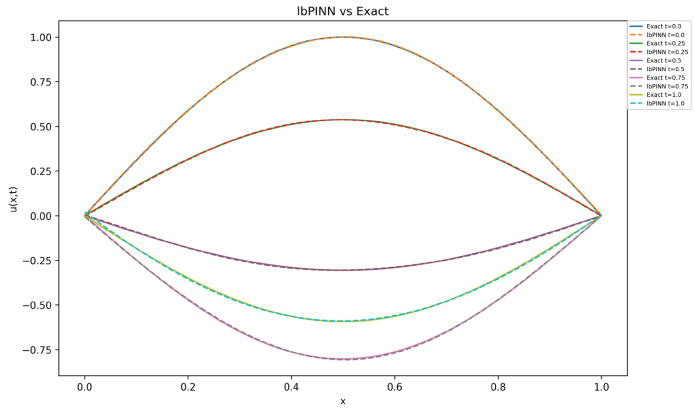
初始条件的损失：$L_{IC1}=\frac{1}{N_{i1}}\sum_{i=1}^{N_{i1}}\left|u\left(x_i,0\right)-A\sin\left(k\pi x_i\right)\right|^2$。

初始速度的损失：$L_{IC2}=\frac{1}{N_{i2}}\sum_{i=1}^{N_{i2}}\left|u_t\left(x_i,0\right)+\frac{\gamma}{2}A\sin\left(k\pi x_i\right)\right|^2$。

边界条件的损失：$L_{BC}=\frac{1}{N_b}\sum_{i=1}^{N_b}\left[\left|u\left(0,t_i\right)-0\right|^2+\left|u\left(1,t_i\right)-0\right|^2\right]$。



(a)



(b)

Figure 7. (a) Comparison of the numerical solution and the exact solution of the PINN algorithm; (b) Comparison of the numerical solution and the exact solution of the lbPINN algorithm
图 7. (a) PINN 算法数值解与精确解对比图 (b) lbPINN 算法数值解与精确解对比图

图 7 分别展示了 PINN 和 lbPINN 在求解一维电报方程时，不同时刻数值解与精确解的对比情况。通过对比精确解、PINN 算法数值解与 lbPINN 算法数值解的曲线可得出以下结论：两种算法得到的解在整体形态和变化趋势上与精确解基本一致，均呈现典型的双曲函数分布特征。然而在关键区域存在明显差异：PINN 数值解在极值点附近与精确解存在可见偏差，曲线光滑度不足；而 lbPINN 数值解的各条曲线与精确解几乎完全重合，特别是在极值点和梯度变化区域都展现出优异的拟合精度。这一对比结果表明，lbPINN 显著提升了对一维电报方程的求解能力，能够更精确地描述电磁波在传输过程中的动态特性。

Table 4. L2 relative errors of the PINN and lbPINN algorithms for the one-dimensional telegraph equation
表 4. 一维电报方程 PINN 算法与 lbPINN 算法的 L2 相对误差

算法	L2 relative error
PINN	3.1449e−02
lbPINN	5.3313e−03

表 4 清晰地展示了一维电报方程 PINN 算法与 lbPINN 算法的 L2 相对误差。根据表 4 中一维电报方程的 L2 相对误差结果分析，lbPINN 算法相较于 PINN 算法展现出显著的精度提升。具体数据显示，PINN 算法的 L2 相对误差为 3.1449×10^{-2}，而 lbPINN 算法将该误差降至 5.3313×10^{-3}，降幅达到 83.0%。这一结果表明，lbPINN 算法在求解一维电报方程时，能够更精确地捕捉电磁波传播的动态特性。误差降低一个数量级的显著改进，验证了 lbPINN 算法在求解这类包含时间和空间二阶导数的偏微分方程时的有效性和优越性，为电磁场模拟提供了更可靠的数值计算方法。

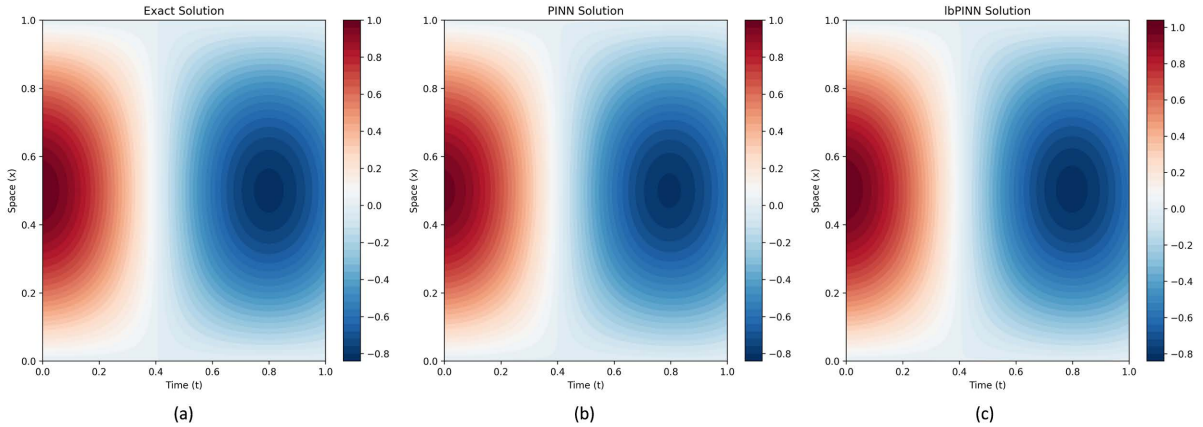


Figure 8. (a) Exact solution; (b) Numerical solution of Algorithm 1 (PINN); (c) Numerical solution of Algorithm 2 (lbPINN)
图 8. (a) 精确解；(b) 算法 1 (PINN)的数值解；(c) 算法 2 (lbPINN)的数值解

基于对一维电报方程数值解与精确解的可视化对比分析，通过观察图 8 的三幅热力图可以得出：lbPINN 算法相较于 PINN 算法在求解精度上取得显著提升。三幅热力图在整体上均呈现从中心区域向四周扩散的同心圆状波动特征，颜色从红色高值区经白色过渡到蓝色低值区，符合电报方程的波动传播规律。然而在细节层面，PINN 数值解在波动前沿出现明显的数值耗散现象，表现为红色区域的边界模糊和能量衰减；而 lbPINN 数值解不仅保持了与精确解高度一致的同心圆形态，其波前锐利度和能量集中度也更接近精确解。这一现象表明，lbPINN 有效抑制了数值耗散，更精确地捕捉了一维电报方程中的波动传播特性，为这类双曲型偏微分方程的求解提供了更可靠的深度学习方案。

4. 结束语

在本文中，我们研究了普通物理信息神经网络(PINN)与自适应损失平衡物理信息神经网络(lbPINN)对于几类偏微分方程的求解以及对比了两种算法的性能。由于 PINN 的性能与收敛性容易受到损失权重多种选择的影响，因此自适应地为其分配合适的权重来组合 PINN 中的多个损失函数至关重要，lbPINN 是一种通过最大化高斯似然并集成可扩展不确定性参数的自适应损失平衡方法，能够同时优化 PINN 中的多重竞争损失项，从而有效提升控制方程的计算精度。我们通过多个数值实验验证了其有效性，将其应用于一维热传导方程、二维泊松方程、一维线性对流方程、一维电报方程的求解，数值实验结果表明，与传统固定权重的 PINN 相比，lbPINN 对于这几类方程的求解在精度上明显提高，降低了误差，数值解与精确解的吻合度显著提高，也提升了稳定性。当前方法在计算效率方面存在一定的局限性，这也是我们未来研究的重点方向。我们计划将 lbPINN 方法扩展至更高维的偏微分方程的求解，并通过优化参数选取策略、结合更高效的优化算法，以克服现有不足，从而推动该方法在更广泛领域中的应用。

基金项目

国家自然科学基金(12161064, 12561043)资助。

参考文献

[1] Avalos, G. and Triggiani, R. (2013) Rational Decay Rates for a PDE Heat-Structure Interaction: Afrequency Domain Approach. *Evolution Equations and Control Theory*, **2**, 233-253. https://doi.org/10.3934/eect.2013.2.233

[2] Ostashev, V.E., Wilson, D.K., Liu, L., Aldridge, D.F., Symons, N.P. and Marlin, D. (2005) Equations for Finite-Difference, Time-Domain Simulation of Sound Propagation in Moving Inhomogeneous Media and Numerical Implementation. *The Journal of the Acoustical Society of America*, **117**, 503-517. https://doi.org/10.1121/1.1841531

[3] Zhang, S., Gu, W., Zhang, X., Lu, H., Yu, R., Qiu, H., *et al*. (2023) Dynamic Modeling and Simulation of Integrated Electricity and Gas Systems. *IEEE Transactions on Smart Grid*, **14**, 1011-1026. https://doi.org/10.1109/tsg.2022.3203485

[4] Abdulla, S.O., Abdulazeez, S.T. and Modanli, M. (2023) Comparison of Third-Order Fractional Partial Differential Equation Based on the Fractional Operators Using the Explicit Finite Difference Method. *Alexandria Engineering Journal*, **70**, 37-44. https://doi.org/10.1016/j.aej.2023.02.032

[5] Taha Abdulazeez, S. and Modanli, M. (2022) Solutions of Fractional Order Pseudo-Hyperbolic Telegraph Partial Differential Equations Using Finite Difference Method. *Alexandria Engineering Journal*, **61**, 12443-12451. https://doi.org/10.1016/j.aej.2022.06.027

[6] Chung, M., Krueger, J. and Liu, H. (2023) Least-Squares Finite Element Method for Ordinary Differential Equations. *Journal of Computational and Applied Mathematics*, **418**, Article 114660. https://doi.org/10.1016/j.cam.2022.114660

[7] Wei, Y., Zhao, Y., Shi, Z., Wang, F. and Tang, Y. (2018) Spatial High Accuracy Analysis of FEM for Two-Dimensional Multi-Term Time-Fractional Diffusion-Wave Equations. *Acta Mathematicae Applicatae Sinica, English Series*, **34**, 828-841. https://doi.org/10.1007/s10255-018-0795-1

[8] Baker, N.A., Alexander, F., Bremer, T., *et al*. (2019) Workshop Report on Basic Research Needs for Scientific Machine Learning: Core Technologies for Artificial Intelligence. USDOE Office of Science (SC), Washington, DC (United States). https://doi.org/10.2172/1478744

[9] Baydin A.G., Pearlmutter, B.A., Radul, A.A., *et al*. (2018) Automatic Differentiation in Machine Learning: A Survey. *Journal of Machine Learning Research*, **18**, 1-43.

[10] 张海斌, 薛毅. 自动微分的基本思想与实现[J]. 北京工业大学学报, 2005, 31(3): 332-336.

[11] Hornik, K., Stinchcombe, M. and White, H. (1989) Multilayer Feedforward Networks Are Universal Approximators. *Neural Networks*, **2**, 359-366. https://doi.org/10.1016/0893-6080(89)90020-8

[12] Raissi, M., Perdikaris, P. and Karniadakis, G.E. (2019) Physics-Informed Neural Networks: A Deep Learning Framework for Solving Forward and Inverse Problems Involving Nonlinear Partial Differential Equations. *Journal of Computational Physics*, **378**, 686-707. https://doi.org/10.1016/j.jcp.2018.10.045

[13] Wang, L. and Yan, Z. (2021) Data-Driven Rogue Waves and Parameter Discovery in the Defocusing Nonlinear Schrödinger Equation with a Potential Using the PINN Deep Learning. *Physics Letters A*, **404**, Article 127408.

https://doi.org/10.1016/j.physleta.2021.127408

[14] Yu, J., Lu, L., Meng, X. and Karniadakis, G.E. (2022) Gradient-Enhanced Physics-Informed Neural Networks for Forward and Inverse PDE Problems. *Computer Methods in Applied Mechanics and Engineering*, **393**, Article 114823. https://doi.org/10.1016/j.cma.2022.114823

[15] D. Jagtap, A. and Em Karniadakis, G. (2020) Extended Physics-Informed Neural Networks (XPINNs): A Generalized Space-Time Domain Decomposition Based Deep Learning Framework for Nonlinear Partial Differential Equations. *Communications in Computational Physics*, **28**, 2002-2041. https://doi.org/10.4208/cicp.oa-2020-0164

[16] Wang, H., Lu, L., Song, S. and Huang, G. (2023) Learning Specialized Activation Functions for Physics-Informed Neural Networks. *Communications in Computational Physics*, **34**, 869-906. https://doi.org/10.4208/cicp.oa-2023-0058

[17] Jagtap, A.D., Shin, Y., Kawaguchi, K. and Karniadakis, G.E. (2022) Deep Kronecker Neural Networks: A General Framework for Neural Networks with Adaptive Activation Functions. *Neurocomputing*, **468**, 165-180. https://doi.org/10.1016/j.neucom.2021.10.036

[18] Lu, L., Meng, X., Mao, Z. and Karniadakis, G.E. (2021) DeepXDE: A Deep Learning Library for Solving Differential Equations. *SIAM Review*, **63**, 208-228. https://doi.org/10.1137/19m1274067

[19] Xiang, Z., Peng, W., Liu, X. and Yao, W. (2022) Self-Adaptive Loss Balanced Physics-Informed Neural Networks. *Neurocomputing*, **496**, 11-34. https://doi.org/10.1016/j.neucom.2022.05.015

[20] 张焕, 张庆, 于纪言. 激活函数的发展综述及其性质分析[J]. 西华大学学报(自然科学版), 2021, 40(4): 1-10.

[21] Jagtap, A.D., Kawaguchi, K. and Karniadakis, G.E. (2020) Adaptive Activation Functions Accelerate Convergence in Deep and Physics-Informed Neural Networks. *Journal of Computational Physics*, **404**, Article 109136. https://doi.org/10.1016/j.jcp.2019.109136

[22] 史加荣, 王丹, 尚凡华, 张鹤于. 随机梯度下降算法研究进展[J]. 自动化学报, 2021, 47(9): 2103-2119.

[23] 董文静, 赵月爱. 基于卷积网络的 Adam 算法的改进[J]. 太原师范学院学报(自然科学版), 2023, 22(3): 5-12.

[24] 赵睿颖. 用于深度学习的一种改进 L-BFGS 算法[J]. 首都师范大学学报(自然科学版), 2021, 42(5): 8-14.

[25] Cipolla, R., Gal, Y. and Kendall, A. (2018) Multi-Task Learning Using Uncertainty to Weigh Losses for Scene Geometry and Semantics. 2018 *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, 18-23 June 2018, 7482-7491. https://doi.org/10.1109/cvpr.2018.00781