基于 PINN 的 Burgers 方程求解模型

报告分类：B

参与人姓名：章瀚元

学号：2233412491

贡献率：100%

签名： 

一、背景简介

1.1 物理信息神经网络（PINN）

物理信息神经网络（physics-informed neural network, PINN）是一种无网格算法，能很好地处理传统数值算法不规则边界条件或者约束条件不足的情况，既不需要离散化处理偏微分方程，又要求较低，得到了广泛应用。目前，PINN 算法已成功应用于不同领域，例如光学、流体力学、系统生物学和生物医学等。在偏微分方程的求解中，PINN 算法对非线性发展方程的求解在非线性科学中具有重要意义。然而 PINN 算法的求解精度过分依赖于训练点数，当训练点数较少时，精度较差，主要原因是随着迭代次数的增加，梯度会消失或者减弱。[1]

1.2 Burgers 方程

解决计算流体力学（Computational Fluid Dynamics, CFD）问题是诸多领域的基本任务，其主要是利用偏微分方程组（Partial Differential Equations, PDEs）对 CFD 的复杂问题进行数值模拟和求解，实现对工程实际问题的定量分析，其应用范围包括天气、气候、车辆以及航空领域发动机的工程设计。Burgers 方程作为流体力学中一类重要的 PDEs，用来描述高速流体运动。该方程不仅在流体力学中得到了广泛地实际应用，也可应用于化学工业、非线性测量声学、动力学等各个领域。它可以理解为是对 CFD 中 Navier-Stokes 方程进行了简化后的数学模型方程，也可以理解为浅水波等问题的数学模型方程。Burgers 方程中主要包含了雷诺数项、对流项、黏度扩散项和时间项等。其中，雷诺数项的作用主要是为了使对流项和黏度扩散项达到平衡。在带有时间项的 Burgers 方程中，通常会出现激波现象，这不仅取决于大雷诺数，并且随着时间的持续增长，对流项将会起着主导作用。长期以来，激波现象是 Burgers 方程进行模拟的一个难点，因此，无论从实践还是理论层面上，对 Burgers 方程的数值方法进行研究具有显著意义，该方程的求解也吸引着众多研究者。经典的数值求解方法主要包括有限体积法、有限元法和有限差分法。虽然这些方法保证了数值分析的精确性，但却十分依赖于网格剖分和初始边界条件及边界区域，一旦这些条件受到影响将改变效果的好坏，此外，还需要昂贵的计算开销。[2]

1.3 神经网络求解物理模型

随着科学计算领域的快速发展，深度学习在图像处理、自然语言等众多领域得到了广泛应用。已有研究证明，深度神经网络与函数拟合有异曲同工之妙。通过网络训练，模型在一个复杂的系统中学习到潜在的特征，并将结果拟合映射出来。在人工智能（Artificial Intelligence, AI）求解 PDEs 研究中，依赖于神经网络的万能逼近定理，在不考虑神经元的条件下，对具有线性输出层和具有激活函数的隐藏层，前馈神经网络能以任意精度拟合任意复杂度的连续函数，这为求解 PDEs 提供了条件。最流行的方法是以纯数据驱动为主，先获得一个数据集并定义好一个算法结构，从中学习到一个具体的模型，这是具有“黑盒”性质的深度神经网络（Deep Neural Network, DNN），其结果好坏与训练数据息息相关，并且缺乏一定的泛化能力，只能针对某个算例。此外，该方法通常需

要大量的模拟数据，这不仅会占用大量的计算资源，而且在实际应用中，高精度的大数据通常是难以获得的。

在科学和工程领域，已经存在以基本物理定律形式存在的知识体系。DNN最近的一个变种，叫做物理信息神经网络（Physical Information Neural Networks, PINNs)[8]，是将自然法则融入 DNN。源于早期的以 PDEs 形式描述的物理模型驱动，将物理模型融合到数据驱动中建立了以物理信息为约束的网络，使得网络能遵循自然法的不变性、对称性和守恒性等物理性质。物理信息在流体力学问题中同样存在数据之间隐含规律复杂多变的客观现象。数据里往往存在控制方程形式的约束，通过将约束条件融入网络，使其从给定的先验信息（数据）中找到正确的 PDEs 常数或物理参数（雷诺数)，实现 PDEs 的精确求解及重要参数（雷诺数）预测，同时具有较好的泛化性能。

作为一类求解 PDEs 方法，PINN 也被用来解决流体力学问题。本文应用 PINN 模型原理，针对数据驱动的神经网络缺乏可解释性问题，提出了一种基于 PINN 的 Burgers 方程求解模型，该模型通过构造两个神经网络，在损失函数中加入了对方程的约束，从而增加其的物理性。[2]

本文主要通过对文献[2]的学习，借助 Pytorch 独立设计具体算法，对 Burgers 方程做了正确性验证实验,并将数值解与解析解进行对比。

二、模型建立

2.1 Burgers 方程

考虑 1+1 维 Burgers 方程:

$$\begin{cases} \frac{\partial u}{\partial t}+\lambda u\frac{\partial u}{\partial x}=\nu\frac{\partial^2 u}{\partial x^2}, & x\in(-1,1), t\in(0,1), \\ u(0,x)=-\sin(\pi x), & x\in[-1,1], \\ u(t,1)=u(t,-1)=0, & t\in[0,1] \end{cases} \tag{1}$$

式中：u 表示流体的速度；v 表示运动粘度；λ 是微分算子的待定系数；x 表示空间坐标，x∈[-1,1];t 表示时间，t∈[0,1]。

取 λ=1.0 和 v=0.01／π。

2.2Burgers 方程的 PINN 模型

2.2.1 选择网络架构

PINN 模型由两种基本算法组成：深度神经网络和自动微分技术。深度神经网络由输入层和输出层之间多个层形成。在 PINN 模型中，一个用于反向传播的前馈神经网络通常使用的是基于链规则的导数技术。并且，信息只从输入层向前移动，在隐藏层的训练下给到输出层。

选用 7 层全神经网络，每层包含 100 个神经元，激活参数为 tanh。tanh 是

双曲切线非线性激活函数。该网络能处理输入的非线性关系，并生成较为平滑的解。

2.2.2 定义损失函数

损失函数是 PINN 的关键部分。它由两个主要组成部分构建：物理损失和网络损失。

物理损失，指量化模型预测与物理方程残差的差异，确保预测遵循已知的物理定律。

$$L_f = \frac{1}{n}\sum_{i=1}^{n}(u_{pred}^{i} - u_{true}^{i})^2, \tag{2}$$

其中，u_{pred}^{i}为网络预测值，u_{true}^{i}真实值。

网络损失，指物理方程约束部分量化模型预测与物理方程残差的差异，确保预测遵循狄利克雷边界条件。

$$L_{b1} = \frac{1}{n}\sum_{i=1}^{n}(u(0,x)_{pred}^{i} - u(0,x)_{true}^{i})^2, \tag{3}$$

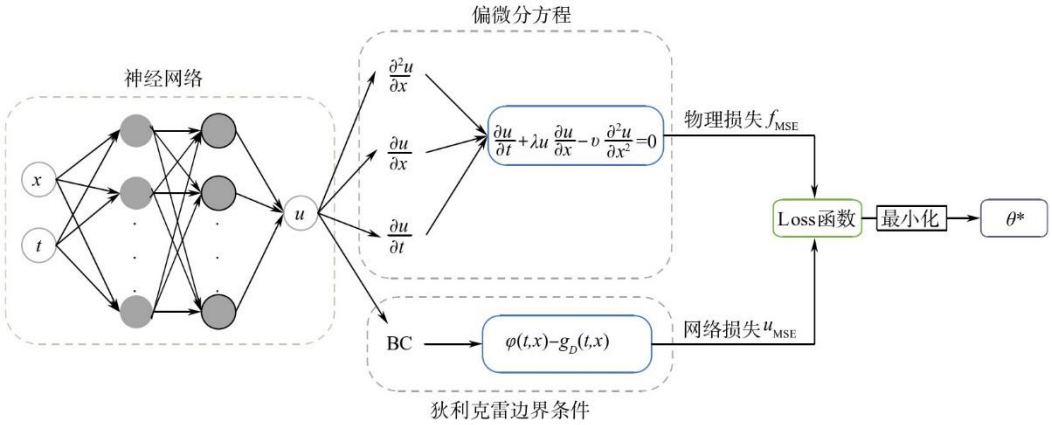
$$L_{b2} = \frac{1}{n}\sum_{i=1}^{n}(u(t,1)_{pred}^{i} - u(t,1)_{true}^{i})^2 + \frac{1}{n}\sum_{i=1}^{n}(u(t,-1)_{pred}^{i} - u(t,-1)_{true}^{i})^2, \tag{4}$$

L_{b1}为初始条件损失，L_{b2}为边界条件损失。

损失函数为：

$$L = \omega_f L_f + \omega_{b1} L_{b1} + \omega_{b2} L_{b2}, \tag{5}$$

图 1 Burgers 方程的 PINN 模型的建立



三、模型求解

本模型采用 adam 优化器进行训练，使用 python 语言编写代码。过程如下：

```
# 初始化模型和优化器
model = PINN()
optimizer = torch.optim.Adam(model.parameters(), lr=1e-4)

# 训练模型
losses = train(model, optimizer, num_epochs=100000)
# 绘制训练损失曲线
plot_loss(losses)
# 绘制数值解图像
plot_solution(model)
#绘制整个 (x, t) 平面的解析解 3d 图
plot_solution_3d()
```

具体代码和函数的实现见附录。

该实验的训练数据集用来训练一个 7 层的深度神经网络，每个隐藏层有 100 个神经元。实验采用 pytorch 框架，训练的总轮数为 100000，共用时 17 分钟。

四、模型分析

4.1 验证模型求解的正确性

损失值为 0.0049（每次训练的损失值不同，这里的损失值和损失曲线为某一次实验的结果），表明模型达到理想效果。

下面是损失值随训练轮数的变化

图 2 100 次训练的损失曲线

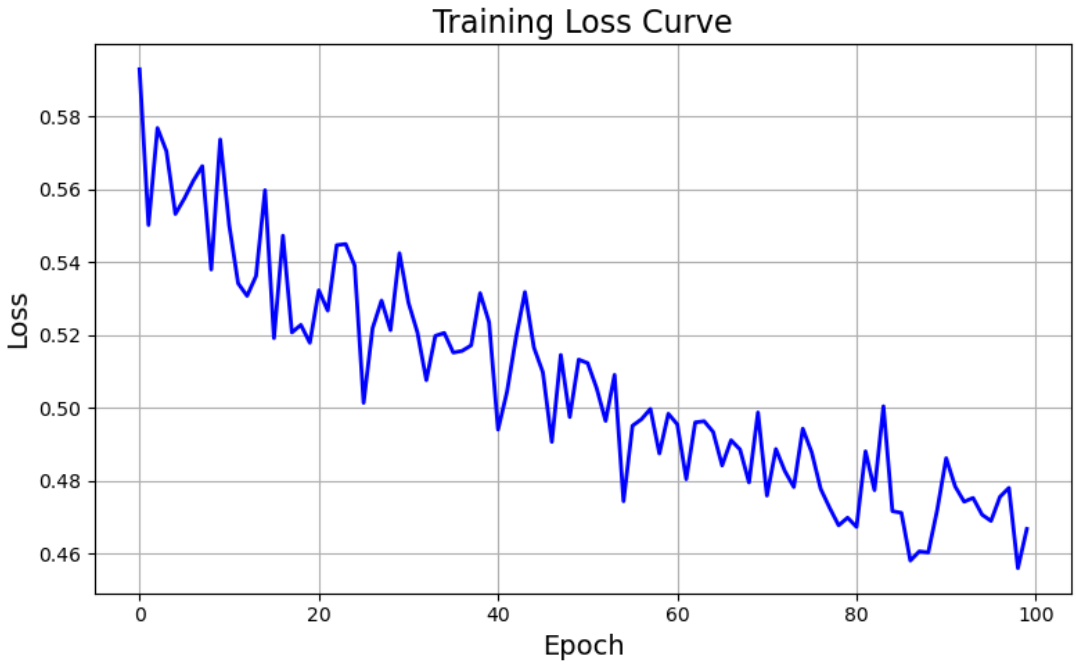
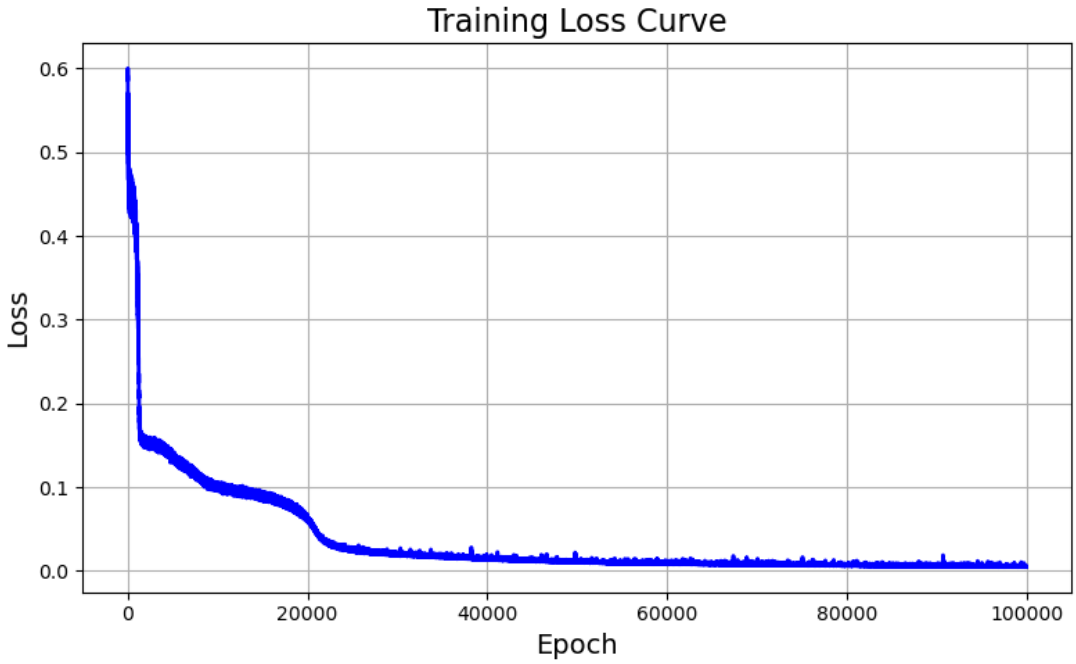


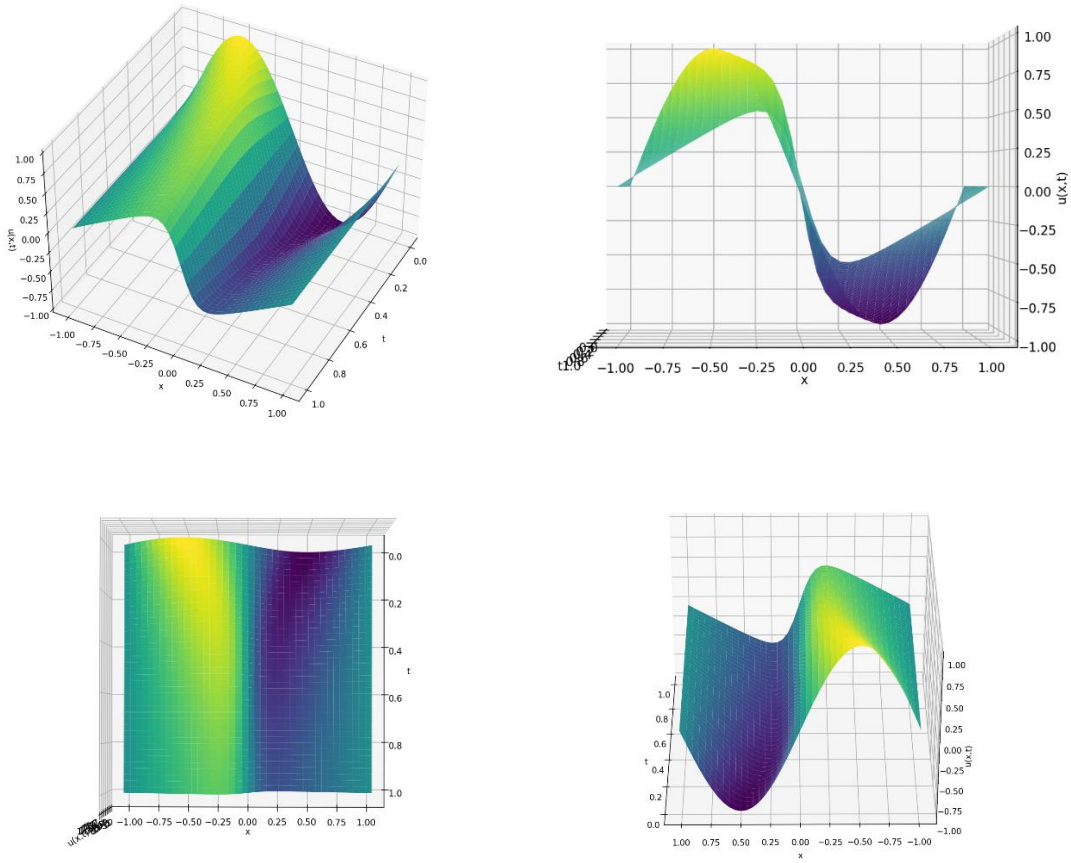
图 3 100000 次训练的损失曲线



4. 2 对比模型解与解析解

解析解的四个角度的三维图如下：

图 4 四个角度下的解析解三维图



本模型在 t=0 时刻对应算法解的时间快照和解析解$u(0,x) = -\sin(\pi x)$对比

图 5 t=0 时刻的模型预测解

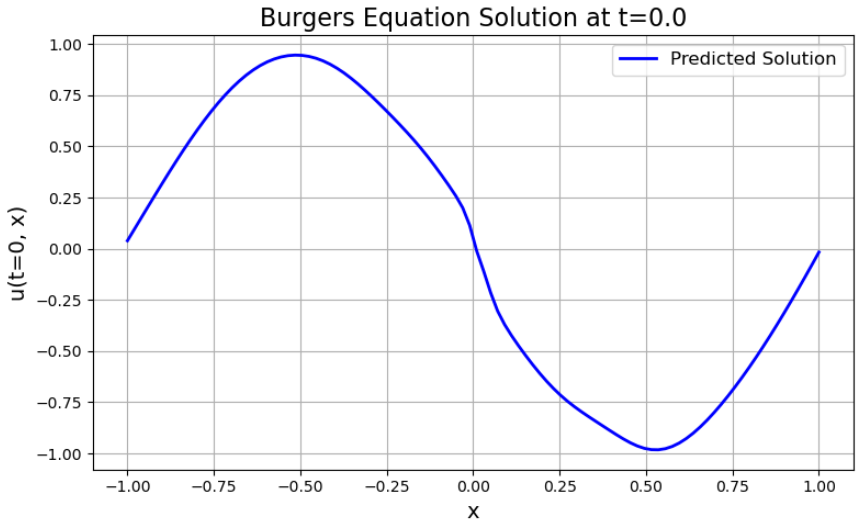


图 6 t=0 时刻的解析解

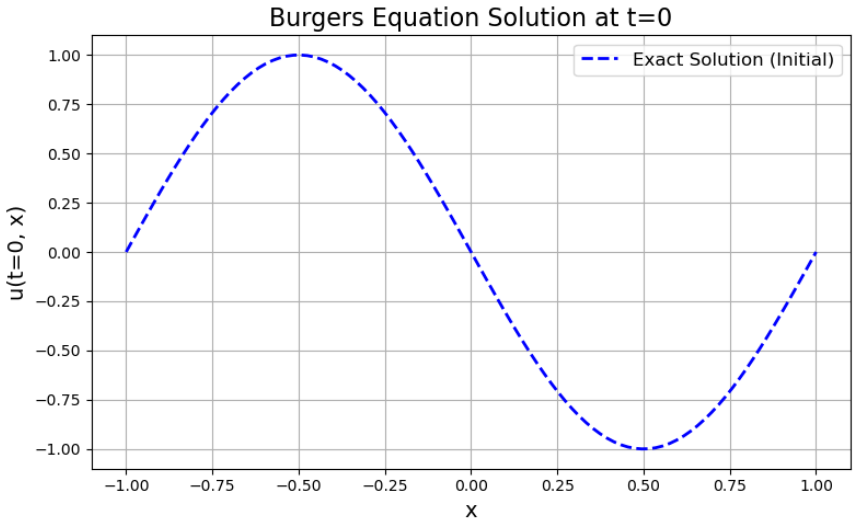


图 7 展示了文献[2]在 4 个时间点，分别是 t=0.20、t=0.40、t=0.60 和 t=0.80 时刻对应算法解的时间快照和真实值，图 8 展示了本模型在 t=0.20、t=0.40、t=0.60 和 t=0.80 时刻对应算法解的时间快照。

图 7 t 在不同时刻的解析解和文献[2]的模型预测解

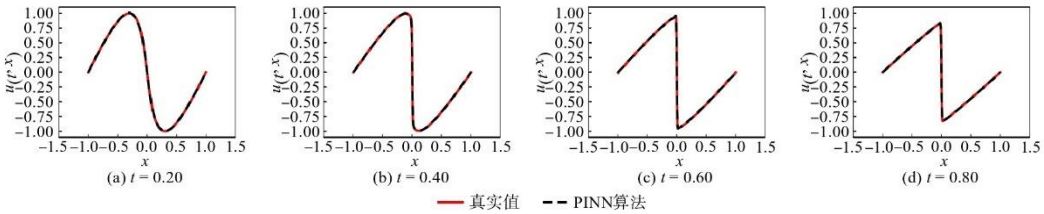
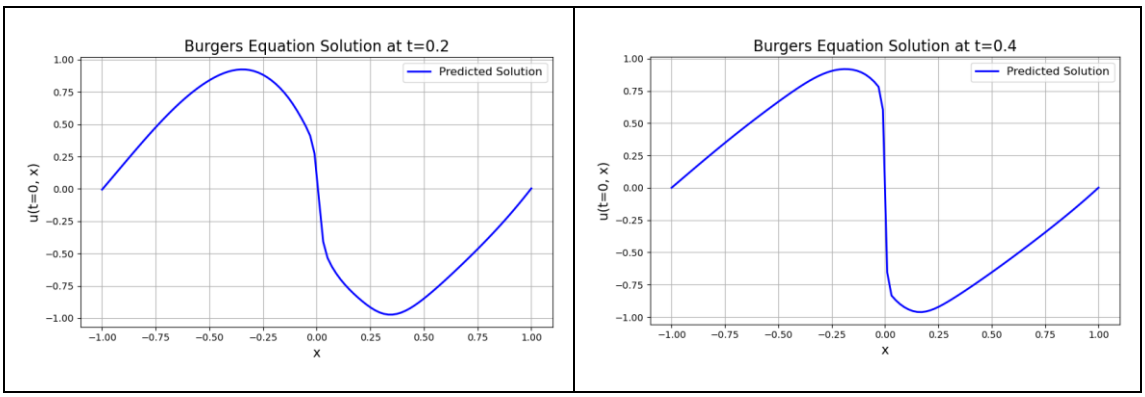
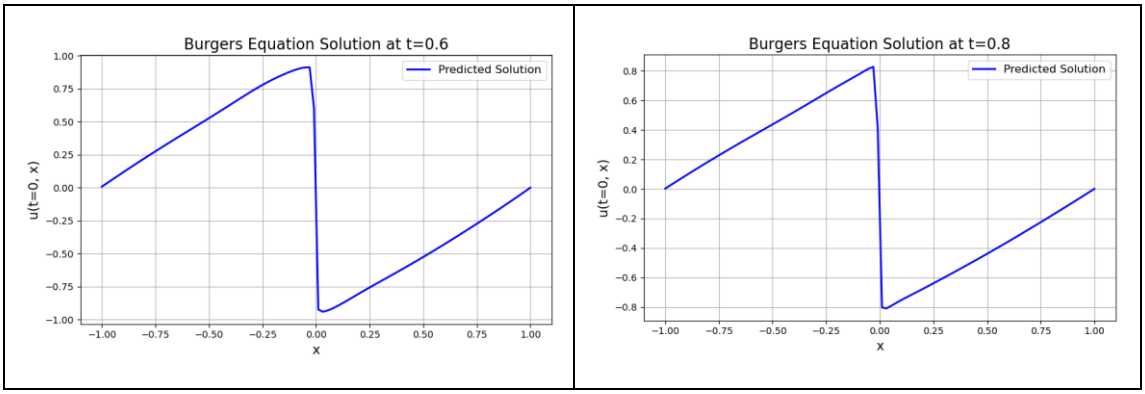


图 8 t 在不同时刻的模型预测解





通过对比，本模型的数值解与解析解相近，但不如文献[2]模型的解准确。

五、结论

本文介绍了如何通过物理信息神经网络(PINN)求解经典的 Burgers 方程。通过 PINN 结合自动微分和物理约束，能够高效求解 Burgers 方程这样的非线性偏微分方程。PINN 不仅能够逼近未知的数值解，还能够通过物理一致性约束增强解的合理性和物理解释性。相比传统的数值方法，PINN 具有处理高位问题和复杂边界条件的潜力。

实验结果表明，该模型精确地求解了 1+1 维 Burgers 方程。本人深入学习并实现了该模型，取得了良好的效果，但仍有不足，例如没有考虑提高训练次数是否能进一步减小误差，如何设计 PINN 解决高维方程。

声明

本人郑重声明，该论文是由本人独立完成的研究成果，没有抄袭其他小组的报告，也没有与任何人讨论过该论文。本论文中所有实验数据和结果均真实可靠，无任何伪造、篡改或剽窃行为。本论文是通过自己设计算法对文献[2]中具体问题的学习和实现，因此有大量内容与文献[2]相关，但模型建立和问题解决的过程与文献[2]不同。

六、参考文献

[1]栗雪娟,刘瑜欣.基于 PINN 及其改进算法求解 KdV-mKdV 方程[J].浙江大学学报(理学版),2024,51(06):702-711.

[2]骆炜杰,李芳,陈鑫.基于 PINN 的 Burgers 方程求解模型[J].信息工程大学学报,2024,25(03):323-330.