



电子科技大学  
University of Electronic Science and Technology of China

求 / 实 / 求 / 真      大 / 气 / 大 / 为

# 内嵌物理知识神经网络 Physics Informed Neural Network ( PINN )

---

汇报人 : xx

时间 : 2023.xx.xx



# 目录/Contents

1. 研究背景
2. 算法原理
3. 求解实验
4. 应用及局限性



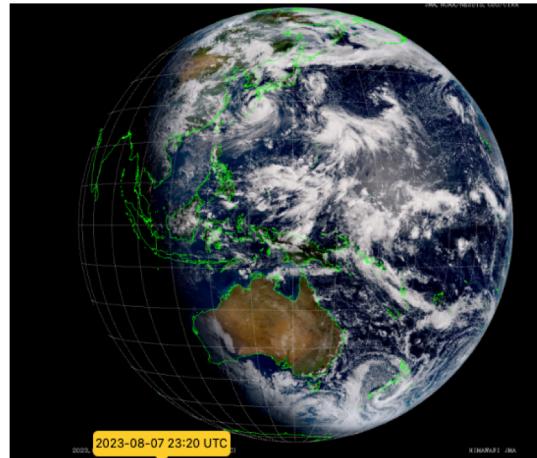
# 01 研究背景



# 研究背景

问题提出：如何建模和预测多物理和多尺度系统的动力学？

例如，地球系统是一个独特复杂的系统，其动力学受到时空尺度上发生的物理、化学和生物过程相互作用的复杂控制等。



传统方法：有限差分、有限元、光谱以及无网格方法数值求解偏微分方程（PDE）



- 1、非线性多尺度系统带来**高昂成本**和**多种不确定性来源**；
- 2、求解**逆问题**（如，推断功能材料中的材料特性）**代价高昂**，需要复杂公式、新算法和代码；
- 3、无法解决具有**缺失、间隙或噪声**边界条件的实际物理问题；

需要一种普遍接受的建模和预测方法，应对现有大量的（具有时空变异性）观测数据，同时可以满足现实世界的物理规则。

# 研究背景

深度学习  
机器学习

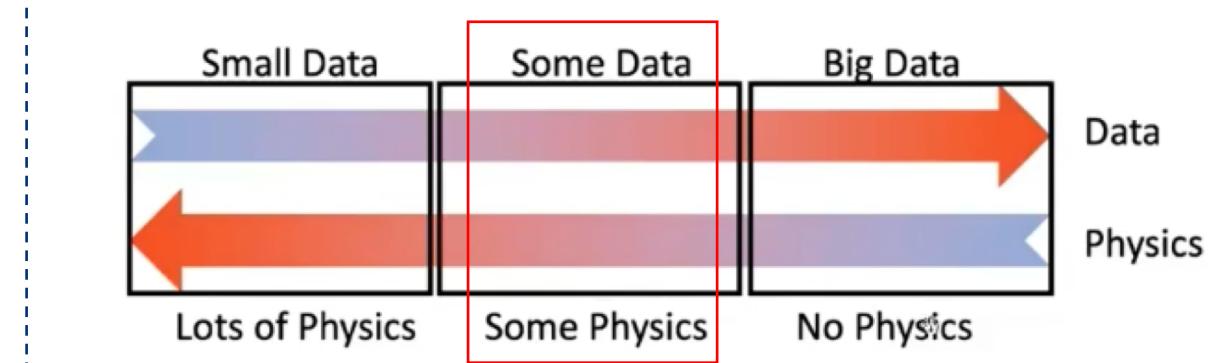


自动提取  
数据特征



- 1、多数ML方法目前都**无法从数据洪流中提取可解释的信息和知识**；
- 2、由于观测数据偏差，预测可能**在物理上不一致或不可信**，导致较差的泛化性能；
- 3、得到的结果大部分不满足物理知识；

提供物理规则（或信息先验）等基本物理规律和领域知识优势，从而提高学习算法的性能——**内嵌物理知识神经网络（PINN）**



- 1、左侧（经典方法）：小数据，了解所有物理知识；
- 2、中间（PINN）：部分数据，缺少一些参数值或偏微分方程的部分项（求正、反问题，最常见）；
- 3、右侧（系统验证/发现）：完全数据驱动，不含任何物理知识（DeepONet）；

- 1、**应对不完美的模型和噪声数据**：PINN公式具有平衡性、规则性；无需昂贵的网络生成；
- 2、**小数据领域上具有强泛化性**：嵌入物理知识从而约束/惩罚深度学习模型，减少高精度必要训练数据的数量；
- 3、**有助于理解深度学习模型**：阐明深度学习背后的内在机制；
- 4、**处理高维度数据**：深度学习模型学习高维度数据的样本分布，用于精细分辨率的图像分类、语言建模和高维PDE求解等问题；
- 5、**量化不确定性**：可靠预测多尺度和多物理系统的演化，用于解决物理系统（如参数的随机性）、数据（噪声）、模型（难以量化）的不确定性；



# 02 算法原理



通过使用自动微分将偏微分方程嵌入神经网络的损失函数，PINN无缝集成了来自测量和偏微分方程PDE的信息。

考虑解决以下PDE问题：

$$u_t + N(u; \lambda) = 0$$

其中 $u(t, x)$ 是需要求的潜在解， $N(u; \lambda)$ 是关于这个解的非线性操作算子（如偏导数）， $\lambda$ 是待定参数。

那么现在存在两种情况：

- 1、参数 $\lambda$ 已知时，如何求未知解 $u(t, x)$ ；
- 2、参数 $\lambda$ 未知时，如何求解 $u(t, x)$ 的同时确定参数 $\lambda$ ；

根据上述公式得到：

$$f(t, x) := u_t + N(u; \lambda)$$

PINN使用神经网络来逼近 $u(t, x)$ 和 $f(t, x)$ ，且这两个网络共享参数（基于神经网络自动微分的功能，链式法则）

# PINN网络框架

具体操作：

首先定义一个网络来拟合 $u(t, x)$ ，然后 $f(t, x)$ 可接在 $u(t, x)$ 后面再加上一些操作算子。

定义损失函数：

$$MSE = MSE_u + MSE_f$$

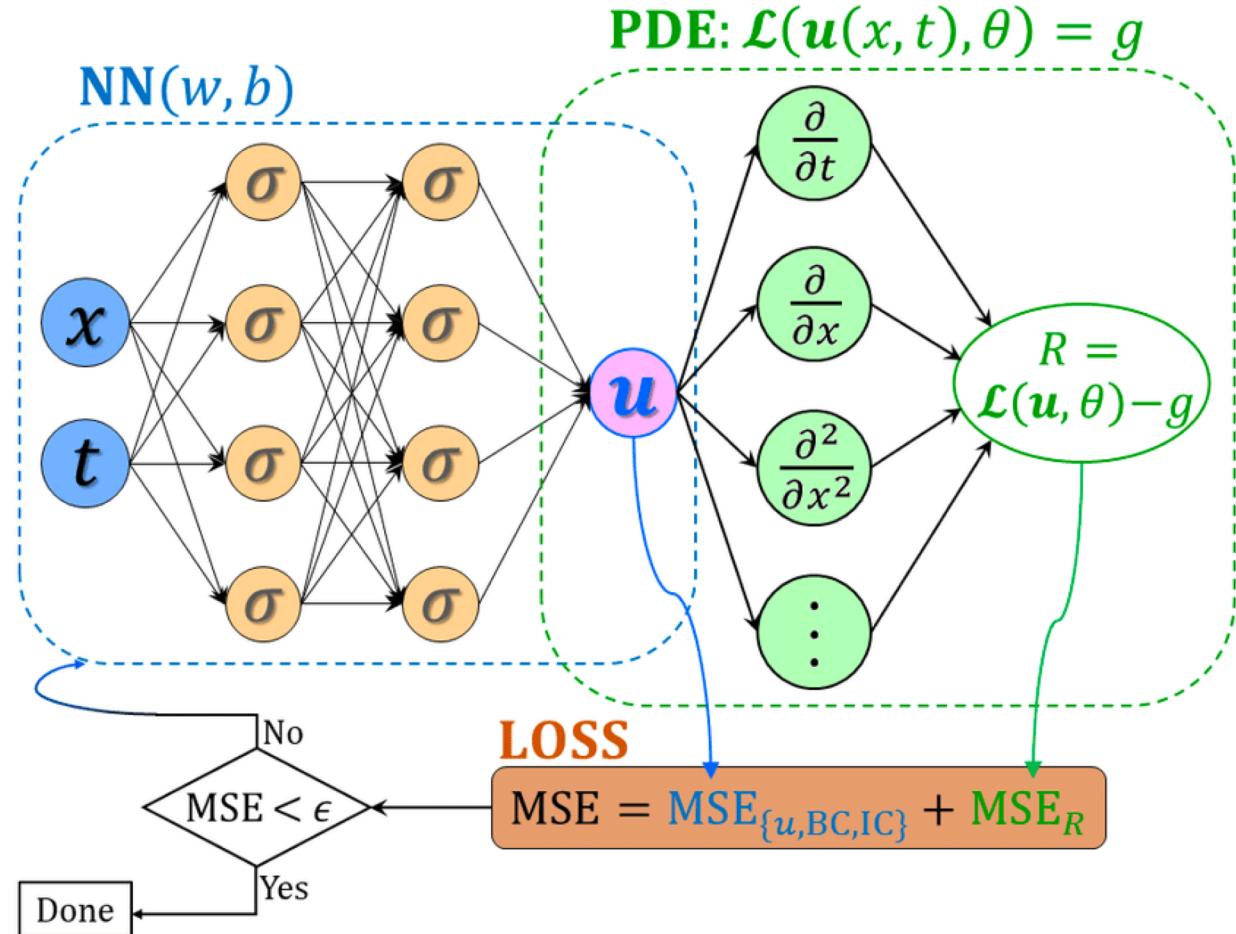
$$MSE_u = \frac{1}{N_u} \sum_{i=1}^{N_u} |u(t_u^i, x_u^i) - u^i|^2$$

$$MSE_f = \frac{1}{N_f} \sum_{i=1}^{N_f} |f(t_f^i, x_f^i)|^2$$

$MSE_u$ （数据损失函数）表示在初始和边界条件处神经网络的拟合值 $u(t_u^i, x_u^i)$ 与真实值的均方误差；

$MSE_f$ （物理损失函数）表示的是神经网络与真实物理规律的均方误差。

当LOSS趋近于0时，我们可以得到PDE的精确解。



# 如何嵌入物理知识——学习

一阶非线性Schrödinger方程如下：其中 $h(t, x)$ 为复数函数， $u$ 是实部， $v$ 是虚部

$$ih_t + 0.5h_{xx} + |h|^2h = 0, \quad x \in [-5, 5], \quad t \in [0, \pi/2] \text{ (PDE)}$$

$$h_t = h(t, x) = u(t, x) + iv(t, x)$$

$$\begin{aligned} h(0, x) &= 2 \operatorname{sech}(x), \\ h(t, -5) &= h(t, 5), \\ h_x(t, -5) &= h_x(t, 5), \end{aligned}$$

$$f(t, x) := ih_t + 0.5h_{xx} + |h|^2h$$

边界条件

$$MSE_0 = \frac{1}{N_0} \sum_{i=1}^{N_0} |h(0, x_0^i) - h_0^i|^2,$$

$$MSE_b = \frac{1}{N_b} \sum_{i=1}^{N_b} (|h^i(t_b^i, -5) - h^i(t_b^i, 5)|^2 + |h_x^i(t_b^i, -5) - h_x^i(t_b^i, 5)|^2)$$

$$MSE_f = \frac{1}{N_f} \sum_{i=1}^{N_f} |f(t_f^i, x_f^i)|^2$$

$$MSE = MSE_0 + MSE_b + MSE_f$$

损失函数定义：

$MSE_0$ 是初始条件损失函数， $MSE_b$ 是周期条件损失函数， $MSE_f$ 是偏微分方程构造的损失函数，当损失函数趋于0时，可以得到PDE较为精确的结果。

# 求解一阶非线性薛定谔方程

损失函数具体形式如下：

$$l = l_1 + l_2 + l_3 + l_4 + l_5 + l_6 + l_7 + l_8$$

其中：

$$l_1 = \frac{1}{N_0} \sum_{i=1}^{N_0} |u(0, x_0^i) - u_0^i|^2$$

$$l_2 = \frac{1}{N_0} \sum_{i=1}^{N_0} |v(0, x_0^i) - v_0^i|^2$$

$$l_3 = \frac{1}{N_b} \sum_{i=1}^{N_b} |u^i(t_b^i, -5) - u^i(t_b^i, 5)|^2$$

$$l_4 = \frac{1}{N_b} \sum_{i=1}^{N_b} |v^i(t_b^i, -5) - v^i(t_b^i, 5)|^2$$

$$l_5 = \frac{1}{N_b} \sum_{i=1}^{N_b} |u_x^i(t_b^i, -5) - u_x^i(t_b^i, 5)|^2$$

$$l_6 = \frac{1}{N_b} \sum_{i=1}^{N_b} |v_x^i(t_b^i, -5) - v_x^i(t_b^i, 5)|^2$$

$$l_7 = \frac{1}{N_f} \sum_{i=1}^{N_f} |u_t + 0.5 * v_{xx} + (u^2 + v^2) * v|^2$$

$$l_8 = \frac{1}{N_f} \sum_{i=1}^{N_f} |v_t + 0.5 * u_{xx} + (u^2 + v^2) * u|^2$$

训练集：

$\{x_0^i, h_0^i\}_{i=1}^{N_0}$   $N_0 = 50$ ，初始数据是从训练数据集中随机解析的 $h(0, x)$ 上的点；

随机取样定位点  $\{t_b^i\}_{i=1}^{N_b}$   $N_b = 50$ ，用于执行周期性边界；

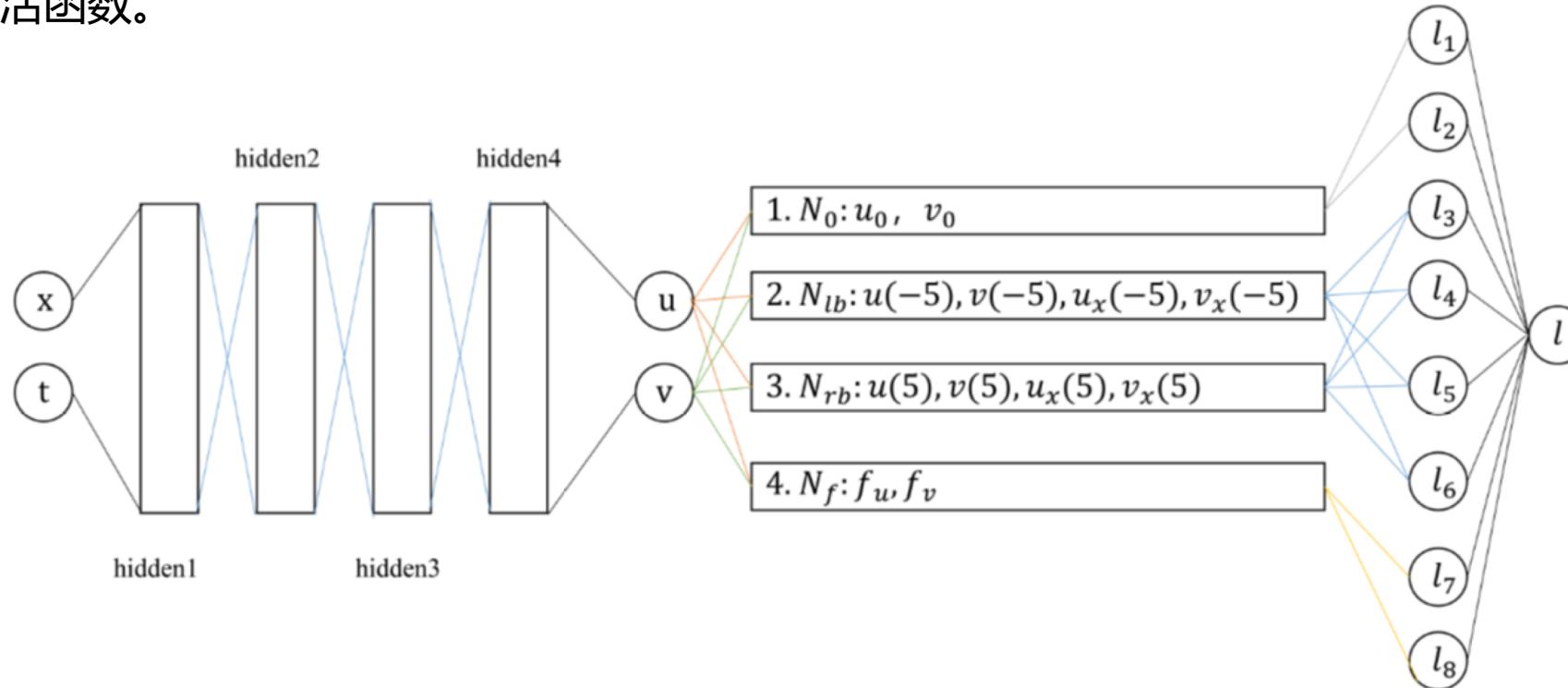
$\{t_f^i, x_f^i\}_{i=1}^{N_f}$   $N_f = 20000$ ，时空域内随机采样点；

$u_0^i, v_0^i$  为谱方法计算出来的真解，其它均为神经网络输出值；

# 一阶非线性薛定谔方程

求解目标：推断薛定谔方程的整个时空解  $h(t, x)$

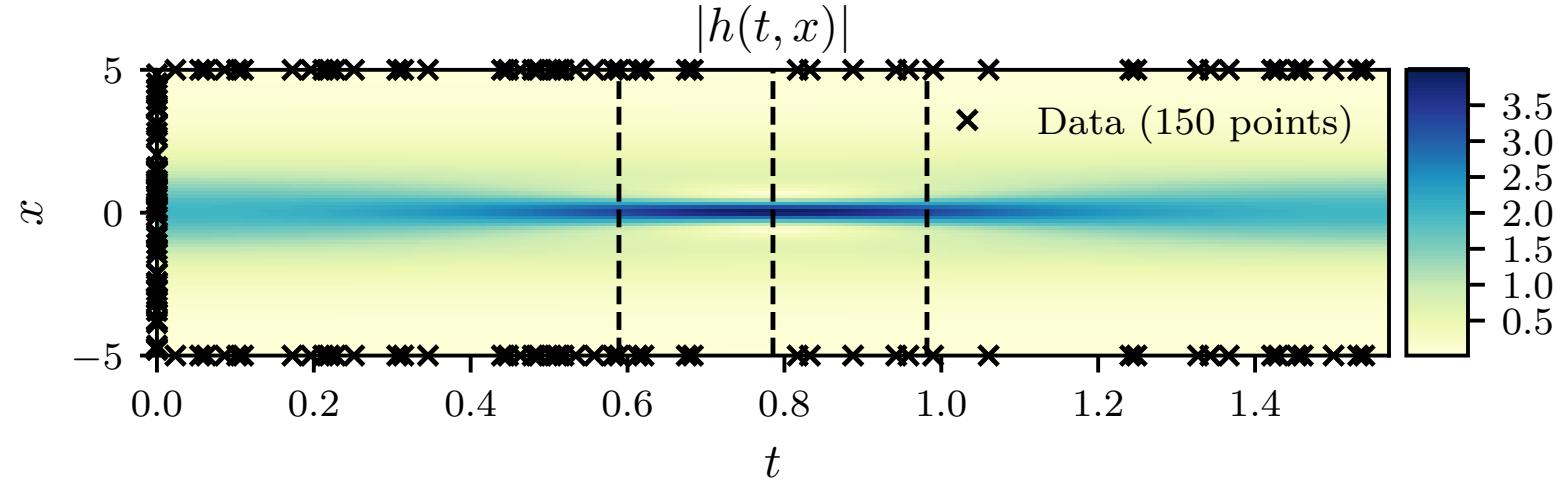
神经网络架构如下：使用5层神经网络来描述潜在函数  $h(t, x) = [u(t, x), v(t, x)]$ ，每层100个神经元，使用双曲正切激活函数。



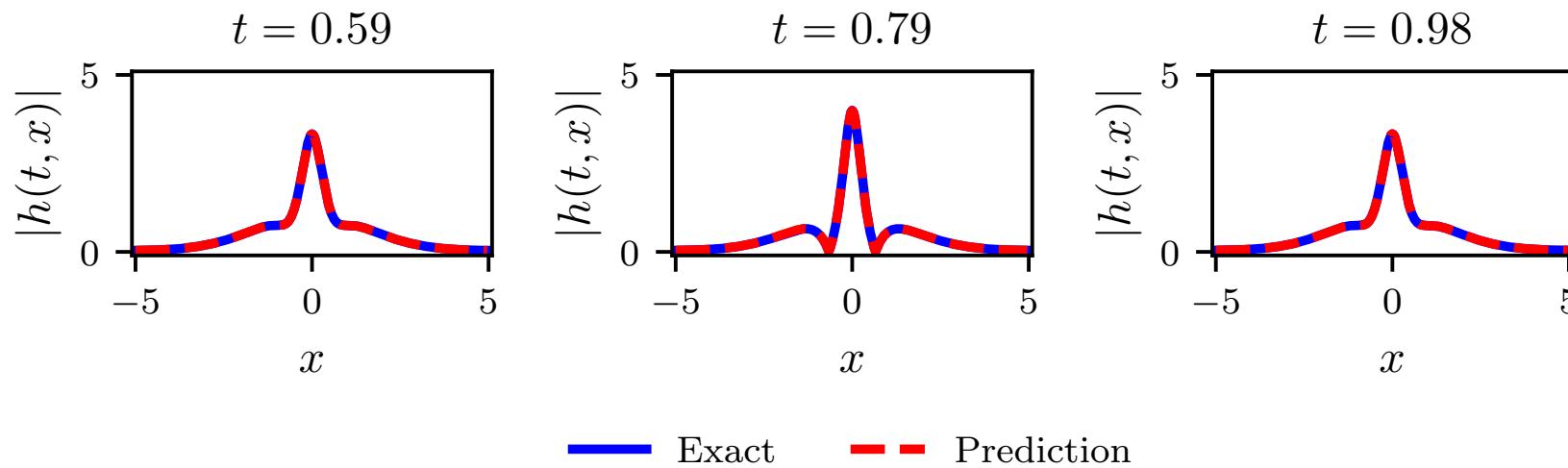
控制输入-输出之间的网络参数训练过程，使其生成结果与输入变量之间尽量保持一定的物理关系

# 一阶非线性薛定谔方程

训练50000次后的输出结果如下：其中 $|h(t, x)| = \sqrt{u^2(t, x) + v^2(t, x)}$



可以看到仅使用少量初始数据，物理信息神经网络就能准确捕捉薛定谔方程错综复杂的非线性行为。





# 03 求解实验



# 一维Burgers方程

用PINN解决由随时间变化的非线性偏微分方程描述的更具挑战性的动态问题。  
在一维上的Burgers方程与边界条件如下：

$$u_t + uu_x - (0.01/\pi)u_{xx} = 0, \quad x \in [-1, 1], \quad t \in [0, 1],$$

$$u(0, x) = -\sin(\pi x),$$

$$u(t, -1) = u(t, 1) = 0.$$

1、**定义** $f(t, x)$ **方程**： $f := u_t + uu_x - (0.01/\pi)u_{xx}$

代码**定义**：

```
def f(t, x):
    u = u(t, x)
    u_t = tf.gradients(u, t)[0]
    u_x = tf.gradients(u, x)[0]
    u_xx = tf.gradients(u_x, x)[0]
    f = u_t + u*u_x - (0.01/tf.pi)*u_xx
    return f
```

2、**通过深度神经网络逼近** $u_t$ ，即 $u(t, x)$ 。

代码**定义**：

```
def u(t, x):
    u = neural_net(tf.concat([t,x],1), weights, biases)
    return u
```

# 一维Burgers方程

3、设计神经网络损失函数： $MSE = MSE_u + MSE_f$

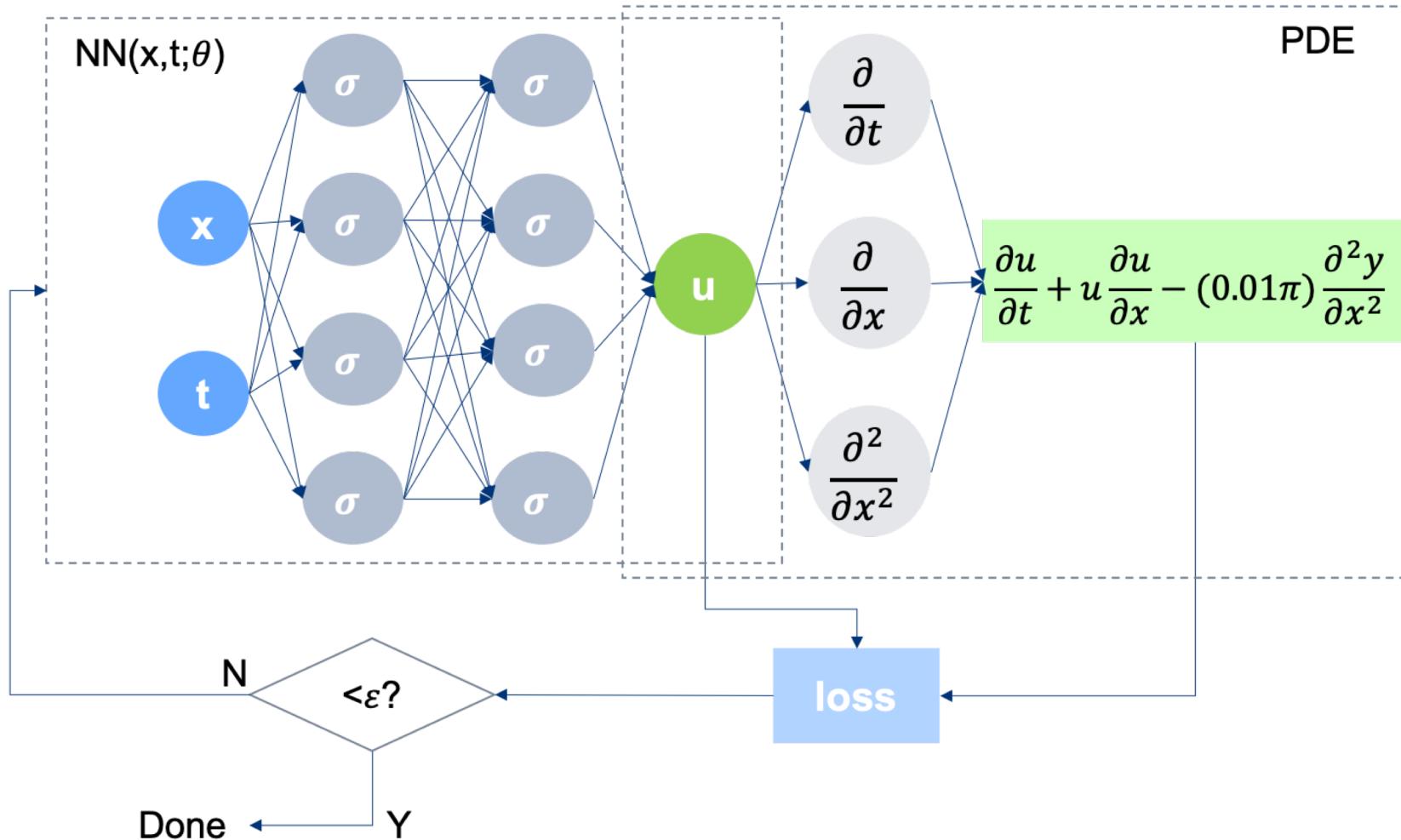
$$MSE_u = \frac{1}{N_u} \sum_{i=1}^{N_u} |u(t_u^i, x_u^i) - u^i|^2$$

$$MSE_f = \frac{1}{N_f} \sum_{i=1}^{N_f} |f(t_f^i, x_f^i)|^2$$

$\{t_u^i, x_u^i, u^i\}_{i=1}^{N_u}$  表示  $u(t, x)$  的初始和边界训练数据， $\{t_f^i, x_f^i\}_{i=1}^{N_f}$  是  $f(t, x)$  方程的指定配置点。

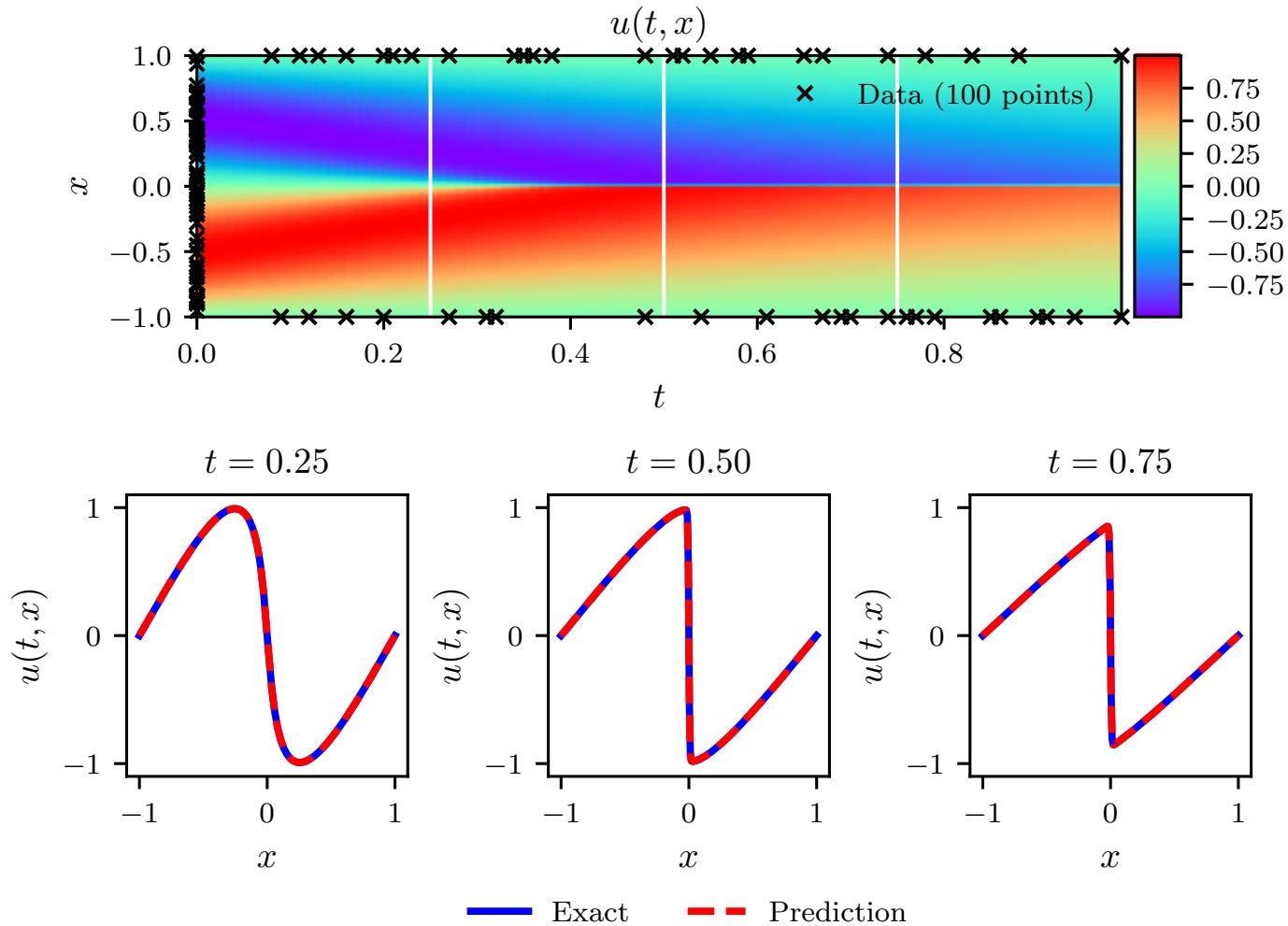
损失  $MSE_u$  对应于初始数据和边界数据， $MSE_f$  对应于  $f(t, x)$

# 网络结构



# 一维Burgers方程

4、下图是我们在数据驱动下求解Burgers 方程的结果：





# 04 应用及局限性



应用：用于流体力学、量子化学、材料科学、分子模拟、地球物理学等问题中。

PINN这种方法或者思想可以弥补科学机器学习领域中单纯数据驱动的弱点。如果把传统数值格式认为是单纯物理知识驱动，那么PINN或者更广义一点的内嵌物理知识机器学习就是数据驱动与知识驱动的融合方法。

1、**多尺度和多物理问题**：可能很难学习高频函数，多个物理过程可能很难通过一个PINN得以实现（每个物理场分别训练，通过并行的架构结合到一起，学习一个耦合解）；

2、**新算法和计算框架**：训练过程可能不够稳健，无法保证向全局最小值的收敛（设计合适的框架和训练算法等）；

3、**数据**：缺少有价值的实验数据集，以及生成这些数据集所需要的物理模型和所有参数（研究人员完善）；

4、**新理论**：缺少一种新理论，来严格分析物理知识学习的能力和局限性。

总而言之，基于物理知识的学习即使在噪声数据和高维环境中也能无缝地集成数据和数学模型，并能非常有效地解决一般的逆问题。



**THANKS**