# 关于神经网络在全波形反演中的应用

Leon Herrmann*[1]，Tim B¨urchner[1]，、Felix Dietrich[2]
和 Stefan Kollmannsberger[1]

[1]慕尼黑工业大学工程与设计学院计算建模与仿真教席，阿尔西斯大街 21 号，慕尼黑，80 333，德国[2]慕尼黑工业大学计算、信息与技术学院计算机科学中的科学计算教席（SCCS），玻尔兹曼大街 3 号，加兴，85748，德国

arXiv:2303.03260v1 [math.NA] 30 Jan 2023

## 摘要

神经网络近年来在求解反问题方面受到关注。一种突出的方- 法是物理信息神经网络（PINNs），它能够求解正问题和反问题。本文所研究的反问题是全波形反演。本文将物理信息神经网络的性能与经典伴随优化进行比较，重点关注三个关键方面：正向求解器、用于反演场的神经网络试探函数，以及用于基于梯度的最小化的梯度灵敏度计算。从物理信息神经网络出发，逐一对这些关键方面进行调整，直至得出经典伴随优化。结果表明，仅将神经网络用于未知材料场的离散化是有益的；在这种情况下，神经网络产生的重建结果不含经典全波形反演方法中通常出现的振荡伪影。基于这一发现，本文提出了一种混合方法。该方法既利用连续伴随方法进行高效梯度计算，也采用未知材料场的神经网络试探函数。这种新的混合方法在二维和三维示例设置中均优于物理信息神经网络和经典伴随优化。

关键词： 物理信息神经网络、全波形反演、深度学习、伴随状态法、反问题、有限差分法

*leon.herrmann@tum.de，通讯作者

预印本已提交至 *Engineering with Computers*。 2023年3月7日

**符号列表**

$\boldsymbol{x}$ 坐标 $\phi_0$ 声源振幅$t$ 时间 $\boldsymbol{x}_s$ 声源位置$\gamma(\boldsymbol{x}, t)$ 指示函数 $\boldsymbol{m}$ 指标参数化$u(\boldsymbol{x}, t)$ 波场 $u_{\mathcal{M}}(\boldsymbol{x}_i, t)$ 传感器测量值 $\hat{}\gamma(\boldsymbol{x})$ 指标预测 $\mathcal{N}[u, \gamma]$ 偏微分方程参数化 $\hat{}u(\boldsymbol{x}, t)$ 波场预测 $\mathcal{B}[u]$ 边界条件算子$\boldsymbol{\gamma}$ 网格点处的指标 $\mathcal{I}[u]$ 初始条件算子$\boldsymbol{u}$ 网格点处的波场 $N_{\mathcal{M}}$ 传感器数量$A_\gamma(\boldsymbol{\theta}; \boldsymbol{x})$ 指标神经网络 $N_{\mathcal{N}}$ 内部网格点数量$A_u(\boldsymbol{\theta}; \boldsymbol{x})$ 波场神经网络 $N_{\mathcal{B}}$ 边界网格点数量$\boldsymbol{\theta}$ 神经网络参数 $N_{\mathcal{I}}$ 初始网格点数量$C_\gamma(\boldsymbol{c}, \boldsymbol{x})$ 常数试探函数 $\mathcal{L}$ 代价函数 $c_i, \boldsymbol{c}$ 常数试探函数系数 $\mathcal{L}_{\mathcal{M}}$ 测量损失$N_i(x)$ 常数试探函数形函数 $\mathcal{L}_{\mathcal{N}}$ 偏微分方程损失 $F[\boldsymbol{\gamma}, \boldsymbol{x}, t]$ 不可训练前向算子 $\mathcal{L}_{\mathcal{B}}$边界条件损失$\rho(\boldsymbol{x})$ $\rho_0$ 密度 $\mathcal{L}_{\mathcal{I}}$ 初始条件损失$c(\boldsymbol{x})$ $c_0$ 波速 $\boldsymbol{\lambda}$, $\lambda$ 可学习惩罚权重$f(\boldsymbol{x}, t)$ 体积力 $\alpha$ 步长、学习率$\phi(t)$ 正弦突发信号声源 $\epsilon$ 较小的正值 $<< 1$ $n_c$ 周期数 $K_\gamma$ 弗雷歇核$\omega, f_\phi$ 声源频率 $u^\dagger$ 伴随波场

# 1. 引言

## 1.1. 动机

自物理信息神经网络（PINN）出现以来 [1–3], 它们一直受到计算力学科学界的广泛关注。尽管其具有新颖性，已有多篇综述文章相继发表 [4–6]。物理信息神经网络的本质是一种变分方法，其中利用神经网络近似解场，并通过最小化在配点评估的微分方程残差来学习网络的权重和偏置。这种通用表述使得求解大量至少可以部分用微分方程描述的问题成为可能。同时，与经典微分方程求解器的复杂性相比，当前的机器学习框架（如 PyTorch [7]和 TensorFlow [8]）提供了构建物理信息神经网络的通用工具，所需额外代码很少且相对简单。

当可用数据量不足以构建有监督代理模型时，该方法展现出应用潜力，例如 [9–15]中所述的情况。如果将其用作正演求解器，对于许多常见的微分方程，物理信息神经网络的计算成本通常更高，而这些方程采用成熟的正向求解器可以高效求解 [16, 17], ，例如有限元 [18] 或有限差分方法 [19]。然而，在经典方法面临困难的情况下，物理信息神经网络很有前景。可能的应用包括高维微分方程 [20, 21], 代理建模 [22–24], 加速传统求解器 [16], 以及反问题 [25–32]。

本文的创新之处有两点。首先，本文指出了物理信息神经网络用于反问题的优势与不足，并将其与经典全波形反演进行比较，其中连续伴随状态法 [33, 34] 被视为最新水平的参考方法。为此，本文从物理信息神经网络逐步过渡到伴随优化，对两种方法各个组成部分的优势与不足进行了全面考察。研究对象是标量波动方程，该方程在无损检测中应用广泛 [35, 36], ，且近期已在 [25, 26] 中结合物理信息神经网络进行了研究。具体而言，本文将深入考察以下三个方面

（a）将物理信息神经网络用作正向求解器，

(b) 将神经网络作为偏微分方程系数场的试探函数，

(c) 梯度计算。

这种剖析引出了表 1 中总结的方法，这些方法将在第 2 节中详细描述和研究，但不包括引言部分 1.3中介绍的更一般的物理信息神经网络设定。

表1：所研究的方法。

| **方法** | **试探函数** | **梯度计算** |
|---|---|---|
| 物理信息神经网络 (第 1.3 节) | $\hat{\gamma}(\boldsymbol{x}) = A_\gamma(\boldsymbol{\theta}_\gamma; \boldsymbol{x})$<br>$\hat{u}(\boldsymbol{x}, t) = A_u(\boldsymbol{\theta}_u; \boldsymbol{x}, t)$ | $\nabla_{\boldsymbol{\theta}_\gamma}\mathcal{L}$ 使用自动微分 |
| 物理信息神经网络与不可训练正向算子 （第 2.1节） | $\hat{\gamma}(\boldsymbol{x}) = A_\gamma(\boldsymbol{\theta}_\gamma; \boldsymbol{x})$<br>$\hat{u}(\hat{\gamma}, \boldsymbol{x}, t) = F[\hat{\gamma}, \boldsymbol{x}, t]$ | $\nabla_{\boldsymbol{\theta}_\gamma}\mathcal{L}$ 与自动微分 |
| 伴随方法 (章节 2.2) | $\hat{\gamma}(\boldsymbol{x}) = C_\gamma(\boldsymbol{c}, \boldsymbol{x})$<br>$\hat{u}(\hat{\gamma}, \boldsymbol{x}, t) = F[\hat{\gamma}, \boldsymbol{x}, t]$ | $\nabla_{\boldsymbol{c}}\mathcal{L}$ 使用自动微分<br>或 $\nabla_{\boldsymbol{c}}\mathcal{L}$ 使用伴随方法 |
| 混合方法 （章节 2.3） | $\hat{\gamma}(\boldsymbol{x}) = A_\gamma(\boldsymbol{\theta}_\gamma, \boldsymbol{x})$<br>$\hat{u}(\hat{\gamma}, \boldsymbol{x}, t) = F[\hat{\gamma}, \boldsymbol{x}, t]$ | $\nabla_{\hat{\gamma}}\mathcal{L}$ 采用伴随方法<br>以及 $\nabla_{\boldsymbol{\theta}_\gamma}$ $\hat{\gamma}$ 与自动微分结合 |

其中大多数方法自然地表现为从完全基于物理信息神经网络的方法逐步过渡到经典伴随优化的各个步骤。相比之下，本文的第二项创新在于提出了一种混合方法，该方法结合了所研究方法的优势，并取得了更准确的结果。

## 1.2. 全波形反演

全波形反演旨在以无损方式估计试样中的材料分布。图1 给出了一个概念示意图，其中包含矩形区域 $\Omega$ 和一个圆形空洞 $\Omega_V$，该空洞即为待检测对象。通过固定数量的声源发射信号（红色圆圈），并由传感器（黑色圆圈）记录信号的图像，从而识别空洞。

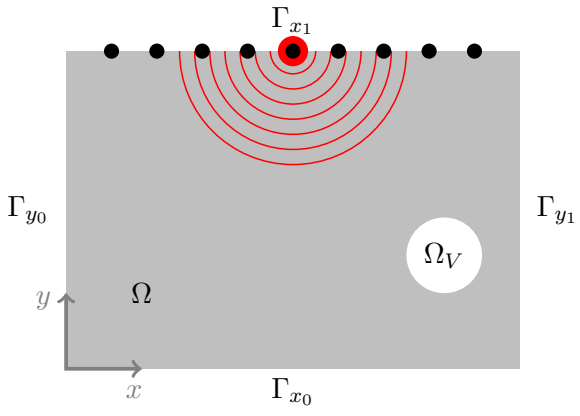



图 1：在含有空洞 $\Omega_V$ 的二维矩形区域 $\Omega$ 上进行超声无损检测。水平边界定义为 $\Gamma_x = \Gamma_{x_0} \cup \Gamma_{x_1}$，垂直边界定义为 $\Gamma_y = \Gamma_{y_0} \cup \Gamma_{y_1}$。

该问题的物理过程可用各向同性非均匀介质中的标量波动方程建模，其中解场 $u(\boldsymbol{x}, t)$、密度 $\rho(\boldsymbol{x})$、波速 $c(\boldsymbol{x})$ 以及

体积力 $f(\boldsymbol{x},\ t)$ 表示为

$$\rho(\boldsymbol{x})u_{tt}(\boldsymbol{x},t) - \nabla\cdot(\rho(\boldsymbol{x})c(\boldsymbol{x})^2\nabla u(\boldsymbol{x},t)) - f(\boldsymbol{x},t) = 0 \qquad \text{on } \Omega\times\mathcal{T}. \tag{1}$$

此外，还规定了齐次诺伊曼边界条件和齐次初始条件：

$$u_x(\boldsymbol{x},t) = 0 \qquad \text{on } \Gamma_y\times\mathcal{T} \tag{2}$$
$$u_y(\boldsymbol{x},t) = 0 \qquad \text{on } \Gamma_x\times\mathcal{T} \tag{3}$$
$$u(\boldsymbol{x},0) = u_t(\boldsymbol{x},0) = 0 \qquad \text{on } \Omega \tag{4}$$

通过缩放密度，利用指示函数 $\gamma$ 对逆问题进行参数化，可最准确地重建空洞和未知的齐次诺伊曼边界。完整材料的密度为常数 $\rho_0$，且 $\gamma$ 受限于 $[\epsilon,1]$之间，其中 $\epsilon \ll 1$ 是一个非零下界：

$$\rho(\boldsymbol{x}) = \gamma(\boldsymbol{x})\rho_0 \tag{5}$$

相比之下，假设波速为常数 $c(\boldsymbol{x}) = c_0$。采用这种参数化后，标量波动方程的形式为：

$$\gamma(\boldsymbol{x})\rho_0 u_{tt}(\boldsymbol{x},t) - \nabla\cdot(\gamma(\boldsymbol{x})\rho_0 c_0^2\nabla u(\boldsymbol{x},t)) - f(\boldsymbol{x},t) = 0 \qquad \text{on } \Omega\times\mathcal{T}. \tag{6}$$

体积力 $f(\boldsymbol{x},\ t)$在位置 $\boldsymbol{x}_s$发出信号。这通常通过空间狄拉克 delta 函数和源项 $\psi(t)$来建模。

$$f(\boldsymbol{x},t) = \psi(t)\delta(\boldsymbol{x}-\boldsymbol{x}_s) \tag{7}$$

无损检测中一种典型的声源是正弦突发信号，其包含 $n_c$ 个周期，频率为$\omega = 2\pi f_\psi$，振幅为 $\psi_0$，因此

$$\psi(t) = \begin{cases} \psi_0\sin(\omega t)\sin(\frac{\omega t}{2n_c}), & \text{for } 0\le t\le\frac{2\pi n_c}{\omega} \\ 0, & \text{for } \frac{2\pi n_c}{\omega} < t. \end{cases} \tag{8}$$

本文所有示例均采用这种突发信号类型。

反演的目标是重建材料分布，即由 $\boldsymbol{m}$ 参数化的指标 $\hat{}\gamma(\boldsymbol{x};\boldsymbol{m})$，使其能够在正演模拟过程中最准确地复现 $N_{\mathcal{M}}$ 传感器测量值 $u_{\mathcal{M}}(\boldsymbol{x}_i,\ t)$，该模拟会得到 $\hat{}u(\boldsymbol{x},\ t;\boldsymbol{m})$。反演被表述为一个优化问题，其中代价函数是预测值 $\hat{}u$ 与测量值 $u_{\mathcal{M}}$之间的均方误差：

$$\mathcal{L}_{\mathcal{M}}(\hat{\boldsymbol{u}},\boldsymbol{u}_{\mathcal{M}}) = \frac{1}{2}\int_{\mathcal{T}}\int_{\Omega}\sum_{i=1}^{N_{\mathcal{M}}}\Big(\hat{u}(\boldsymbol{x},t;\boldsymbol{m}) - u_{\mathcal{M}}(\boldsymbol{x}_i,t)\Big)^2\delta(\boldsymbol{x}-\boldsymbol{x}_i)d\Omega d\mathcal{T}. \tag{9}$$

$\mathcal{L}_{\mathcal{M}}$ 关于参数 $\boldsymbol{m}$ 的梯度表示为 $\nabla_m\mathcal{L}_{\mathcal{M}}(\hat{}\boldsymbol{u}(\boldsymbol{m})\ \boldsymbol{u}_{\mathcal{M}})$，并通过伴随状态法计算。随后，该梯度用于基于梯度的优化过程，通过迭代更新参数 $\boldsymbol{m}$ 来最小化 $\mathcal{L}_{\mathcal{M}}$：

$$\boldsymbol{m}^{(j+1)} = \boldsymbol{m}^{(j)} - \alpha\nabla_{m^{(j)}}\mathcal{L}_{\mathcal{M}} \tag{10}$$

其中 $\alpha$ 为步长，上标$j$）$=1$、2、.....为迭代计数器。因此，这一迭代过程会得到指标的优化估计值 $\hat{}\gamma(\boldsymbol{x};\boldsymbol{m})$。对于多个声源，迭代更新公式（10）时，必须考虑均方误差 $\mathcal{L}_{\mathcal{M}}$ 及其梯度 $\nabla_m\mathcal{L}_{\mathcal{M}}$ 的平均值。使用伴随方法计算梯度的更多细节见 [37]。关于反问题中伴随方法的更深入说明，请参见 Givoli [33]。

### 1.3. 物理信息神经网络

物理信息神经网络（PINN）的思路是使用人工神经网络 $A$ 近似解 $\hat{}u$，和/或微分方程的系数 $\hat{}\gamma$，采用配点法。在本文中，考虑方程（6），其中解 $\hat{}u$ 为波场， $\rho_0$ 按 $\gamma(\boldsymbol{x})$进行缩放，如方程（5）所示。在最一般的情况下， $u$ 和 $\gamma$ 均由神经网络 $A = (A_u, A_\gamma)$进行近似：

$$\hat{u} = A_u(\boldsymbol{x}, t; \boldsymbol{\theta}_u) \tag{11}$$
$$\hat{\gamma} = A_\gamma(\boldsymbol{x}, t; \boldsymbol{\theta}_\gamma) \tag{12}$$

其中， $\boldsymbol{\theta} = \{\boldsymbol{\theta}_u, \boldsymbol{\theta}_\gamma\}$ 是神经网络的模型参数，帽号表示 $u$ 和 $\gamma$ 的值是神经网络的预测。在本文余下部分，对于仅使用两个神经网络之一（即 $A_u$ 或 $A_\gamma$）的模型，将不带下标的记号 $\boldsymbol{\theta}$ 用作相应神经网络参数的表示。

随后，通过一个与上一节所述经典全波形反演中所用方法类似的优化过程，求解由方程（2）至（4）以及（6）定义的初边值问题。为此，首先将初边值问题写成残差形式：

$$\mathcal{N}[u; \gamma; \boldsymbol{x}, t] = \gamma(\boldsymbol{x})\rho_0 u_{tt}(\boldsymbol{x}, t) - \nabla \cdot (\gamma(\boldsymbol{x})\rho_0 c_0^2 \nabla u(\boldsymbol{x}, t)) - f(\boldsymbol{x}, t) \tag{13}$$
$$\mathcal{B}_x[u; \boldsymbol{x}, t] = u_x(\boldsymbol{x}, t) \tag{14}$$
$$\mathcal{B}_y[u; \boldsymbol{x}, t] = u_y(\boldsymbol{x}, t) \tag{15}$$
$$\mathcal{I}_0[u; \boldsymbol{x}, t] = u(\boldsymbol{x}, t) \tag{16}$$
$$\mathcal{I}_1[u; \boldsymbol{x}, t] = u_t(\boldsymbol{x}, t). \tag{17}$$

算子 $\mathcal{N}$、 $\mathcal{B}_x$、 $\mathcal{B}_y$、 $\mathcal{I}_0$、 $\mathcal{I}_1$ 是相应方程（2）至（4）以及（6）的残差。这些残差在若干配点 $N = N_\mathcal{N} + N_{\mathcal{B}_x} + N_{\mathcal{B}_y} + N_{\mathcal{I}_0} + N_{\mathcal{I}_1}$处计算，所用的是由方程（11）和（12）给出的当前解系数场近似值。这需要计算方程（11）和（12）关于 $\boldsymbol{x}$和 $t$ 的一阶和二阶导数，可通过反向模式自动微分 [38] 或数值微分 [10, 39]来计算。

因此，整体优化过程要最小化的损失泛函，即代价函数$\mathcal{L}$，定义为在配点处各项残差（微分方程残差 $\mathcal{L}_\mathcal{N}$、边界条件残差 $\mathcal{L}_\mathcal{B}$ 和初始条件残差 $\mathcal{L}_\mathcal{I}$）评估值的加权平方和：

$$\mathcal{L}_\mathcal{N}(\boldsymbol{u}, \boldsymbol{\gamma}) = \frac{1}{2N_\mathcal{N}} \sum_{i=1}^{N_\mathcal{N}} \lambda_{\mathcal{N},i} (\mathcal{N}[u_i; \gamma_i])^2 \tag{18}$$
$$\mathcal{L}_\mathcal{B}(\boldsymbol{u}) = \frac{1}{2N_{\mathcal{B}_x}} \sum_{i=1}^{N_{\mathcal{B}_x}} \lambda_{\mathcal{B}_x,i} (\mathcal{B}_x[u_i])^2 + \frac{1}{2N_{\mathcal{B}_y}} \sum_{i=1}^{N_{\mathcal{B}_y}} \lambda_{\mathcal{B}_y,i} (\mathcal{B}_y[u_i])^2 \tag{19}$$
$$\mathcal{L}_\mathcal{I}(\boldsymbol{u}) = \frac{1}{2N_{\mathcal{I}_0}} \sum_{i=1}^{N_{\mathcal{I}_0}} \lambda_{\mathcal{I}_0,i} (\mathcal{I}_0[u_i])^2 + \frac{1}{2N_{\mathcal{I}_1}} \sum_{i=1}^{N_{\mathcal{I}_1}} \lambda_{\mathcal{I}_1,i} (\mathcal{I}_1[u_i])^2 \tag{20}$$
$$\mathcal{L}(\boldsymbol{u}, \boldsymbol{\gamma}) = \lambda_\mathcal{N} \mathcal{L}_\mathcal{N}(\boldsymbol{u}, \boldsymbol{\gamma}) + \lambda_\mathcal{B} \mathcal{L}_\mathcal{B}(\boldsymbol{u}) + \lambda_\mathcal{I} \mathcal{L}_\mathcal{I}(\boldsymbol{u}) \tag{21}$$

为抵消损失项幅值不平衡的问题， $\mathcal{L}_\mathcal{N}$， $\mathcal{L}_\mathcal{B}$， $\mathcal{L}_\mathcal{I}$通常会将每个损失项乘以一个 $\lambda$ 专属于该损失项的权重。惩罚权重可以利用深度视觉中的注意力机制自动计算 [40, 41], ，其中惩罚权重通过相对于代价函数进行最大化来更新 $\mathcal{L}$ ，从而形成极小极大优化过程。基于这一思路进一步扩展，如 [42, 43], 所述，将每个配点处的求值视为一个独立的等式约束，因此为每个配点分配一个惩罚权重，如损失方程（18）、（19）和（20）所示。这一扩展使得能够为其分配

更大的权重，从而更加重视重要性较高的配点，即产生较大残差的点。这将得到如下形式的整体极小极大优化

$$\min_{\boldsymbol{\theta}} \max_{\boldsymbol{\lambda}} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\lambda}), \tag{22}$$

这种极小极大优化将最大化 $\alpha_{\max}$ 的学习率作为最小化 $\alpha_{\min}$学习率之外的另一个超参数。

方程（22）中提出的优化问题通过以下梯度求解： $\nabla_{\boldsymbol{\theta}}\mathcal{L}$ 和 $\nabla_{\boldsymbol{\lambda}}\mathcal{L}$，这些梯度通过反向模式自动微分获得。本文所有示例均使用 Adam 优化器 [44] 作为优化过程。不过，更先进的优化器（例如 Adam 与 L-BFGS 的组合） [45] 通常能实现更快的收敛（例如，参见 [46]）。

将物理信息神经网络应用于正问题和反问题时存在重要差异。这些差异按三类问题归纳于表 2 中：（a）正问题；（b）具有完整域知识的反问题；（c）具有部分域知识的反问题。

表 2：概　　述正问题与反问题在物理信息神经网络表述上的差异　　.

| **问题** | **代价函数** | **神经网络** |
|---|---|---|
| (a) 正问题 | $\mathcal{L}(\hat{\boldsymbol{u}}, \boldsymbol{\gamma})$ $= \mathcal{L}_{\mathcal{N}}(\hat{\boldsymbol{u}}, \boldsymbol{\gamma}) + \mathcal{L}_{\mathcal{B}}(\hat{\boldsymbol{u}}) + \mathcal{L}_{\mathcal{I}}(\hat{\boldsymbol{u}})$ | $\hat{u}_i = A_u(\boldsymbol{x}_i, \ t_i; \boldsymbol{\theta})$ |
| (b)逆问题 (完整域知识) | $\mathcal{L}(\boldsymbol{u}_{\mathcal{M}}, \ \hat{\boldsymbol{\gamma}})$ $= \mathcal{L}_{\mathcal{N}}(\boldsymbol{u}_{\mathcal{M}}, \ \hat{\boldsymbol{\gamma}})$ | $\hat{\gamma}_i = A_\gamma(\boldsymbol{x}_i, t_i, \boldsymbol{\theta})$ |
| (c)逆问题 （部分域知识） | $\mathcal{L}(\hat{\boldsymbol{u}}, \boldsymbol{u}_{\mathcal{M}}, \hat{\boldsymbol{\gamma}})$ $= \mathcal{L}_{\mathcal{M}}(\hat{\boldsymbol{u}}, \boldsymbol{u}_{\mathcal{M}}) + \mathcal{L}_{\mathcal{N}}(\hat{\boldsymbol{u}}, \hat{\boldsymbol{\gamma}})$ $+\mathcal{L}_{\mathcal{B}}(\hat{\boldsymbol{u}}) + \mathcal{L}_{\mathcal{I}}(\hat{\boldsymbol{u}})$ | $\hat{u}_i = A_u(\boldsymbol{x}_i, \ t_i; \boldsymbol{\theta}_u)$ 以及 $\hat{\gamma}_i = A_\gamma(\boldsymbol{x}_i, \ t_i; \boldsymbol{\theta}_\gamma)$ |

**问题类型 (a) - 正问题**：在正问题中，仅使用神经网络$A_u$ （来自方程（11）） 作为波场的近似器 $u(\boldsymbol{x}, t)$，因为 $\gamma(\boldsymbol{x}, t)$ 已知。近似场 $\hat{u}$ 的梯度是相对于输入 $\boldsymbol{x}$ 和 $t$计算的，从而可以计算方程（ $\mathcal{L}(\hat{\boldsymbol{u}}, \boldsymbol{\gamma})$ 中的代价函数（21）。随后，通过执行方程（22中的极小极大优化来求解波场。文献中可以找到用于波动方程正问题的物理信息神经网络变体 [47–50]。

稍作修改后，物理信息神经网络框架也可用于反问题，即识别空间变化的指示函数 $\gamma(\boldsymbol{x})$。

问题类型 (b) - 具有解场全域知识的逆问题 $u(\boldsymbol{x}, t)$：由于 $u(\boldsymbol{x}, t)$ 以测量值 $\boldsymbol{u}_{\mathcal{M}}$ 的形式在各处均已知，因此无需使用神经网络来预测 $A_u$ 。相反，可直接使用 $\boldsymbol{u}_{\mathcal{M}}$ 根据方程 (21) 计算代价函数 $\mathcal{L}$。由于波场已近似满足初始条件和边界条件，代价函数 $\mathcal{L}(\boldsymbol{u}_{\mathcal{M}}, \hat{\boldsymbol{\gamma}})$ 现在仅由微分方程 $\mathcal{L}_{\mathcal{N}}(\boldsymbol{u}_{\mathcal{M}}, \hat{\boldsymbol{\gamma}})$ 的残差项构成。其中，使用神经网络 $A_\gamma$来预测 $\hat{\boldsymbol{\gamma}}$，并将 $u(\boldsymbol{x}, t)$ 直接代入 $\mathcal{L}_{\mathcal{N}}(\boldsymbol{u}_{\mathcal{M}}, \hat{\boldsymbol{\gamma}})$ 作为 $\boldsymbol{u}_{\mathcal{M}}$。关于如何使用物理信息神经网络求解这一逆问题的详细说明见 [25]，其中介绍了最新水平。

问题类型 (c) - 具有解场部分知识的逆问题$u(\boldsymbol{x}, t)$：在无损检测领域，掌握完整的波场知识（即 $\Omega \times \mathcal{T}$中的每个位置）并不常见。通常，接收器只能放置在边界的部分位置，如图1所示，因此只能获得部分知识。然而，为了计算方程 (18) 的微分方程损失 $\mathcal{L}_{\mathcal{N}}$ ，必须知道 $\Omega \times \mathcal{T}$中每个位置的波场 $u(\boldsymbol{x}, t)$和缩放函数 $\gamma(\boldsymbol{x})$。因此，在具有部分域知识的逆问题中，通过使用两个独立的神经网络来近似这两个量，从而计算微分方程损失；也就是说，同时使用方程 (11) 和方程 (12)，如图2所示。代价函数 $\mathcal{L}(\hat{\boldsymbol{u}}, \hat{\boldsymbol{\gamma}})$由边界条件损失（方程 (19)）、初始条件损失（方程 (20)）、微分方程损失 $\mathcal{L}_{\mathcal{N}}(\hat{\boldsymbol{u}}, \hat{\boldsymbol{\gamma}})$（方程 (18)）以及测量损失 $\mathcal{L}_{\mathcal{M}}(\hat{\boldsymbol{u}}, \boldsymbol{u}_{\mathcal{M}})$（方程 (9)）组成。请注意，测量数据 $\boldsymbol{u}_{\mathcal{M}}$ 现在通过

使用正向解ˆ的预测来计算测量损失$\boldsymbol{u}$。例如， [26] 给出了这一物理信息神经网络逆问题的当前最先进方法，并提供了相应的 Python 代码： [51]。

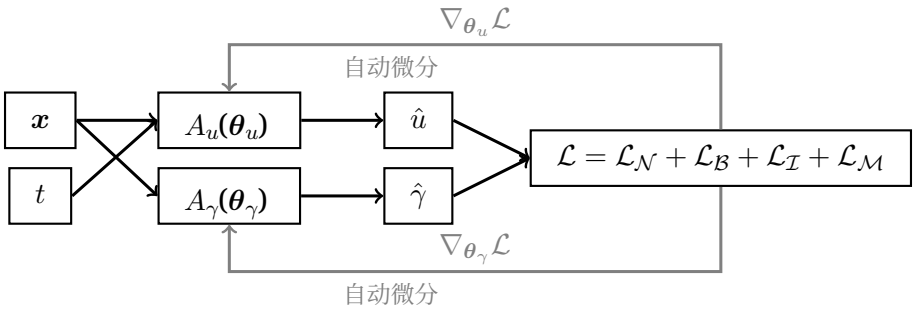



图2：使用标量波动方程进行反演时物理信息神经网络的预测与训练。波场 ˆ$u$ 和指标 ˆ$\gamma$ 的预测通过神经网络 $A_u(\boldsymbol{\theta}_u)$ 和 $A_\gamma(\boldsymbol{\theta}_\gamma)$ 获得。这些网络在训练过程中根据通过自动微分获得的代价函数 $\mathcal{L}$ 的梯度进行更新。

## 2. 神经网络在反演中的作用

为探究物理信息神经网络用于全波形反演的优缺点，我们对物理信息神经网络及其训练流程进行逐步修改，如图 2 所示，直至其与经典伴随优化无法区分。

文献 [25, 26] 讨论了物理信息神经网络在全波形反演中的应用，其中采用了以 $\boldsymbol{x}$ 和 $t$ 为输入的全连接神经网络。相比之下，本文采用卷积神经网络，因为其空间等变性能够显著加快训练速度并加速收敛。这一点已在计算力学等更一般的背景下得到报告，例如 [10, 39] ，且并非本文的主要关注点。此外，这一修改有助于更好地过渡到所介绍的方法并进行比较，也不应视为对物理信息神经网络方法的削弱。尽管存在这一细微差异，我们仍将 [26] 视为当前最新水平下仅使用纯物理信息神经网络和部分域知识所能达到效果的参照案例。

### 2.1. 采用不可训练正向算子的物理信息神经网络 r

图 2 展示了对物理信息神经网络所做的第一项修改：用不可训练的正向算子 $A_u(\boldsymbol{\theta})$ 替换神经网络 $F$ （以 ˆ$\gamma$ 作为额外输入），后者的任务是学习正向解。采用不可训练正向算子的物理信息神经网络的新结构如图 3所示。正向算子可以是任意类型的代理模型，例如预训练神经网络，也可以是经典求解方法，例如有限元方法 [18] 或有限差分方法 [19]。对正向算子的一项重要计算要求 $F$ 是能够基于 GPU 实现；与基于 CPU 的求解器相比，这是预训练神经网络的一项优势。不过，有限差分方法通常也会利用 GPU 加速，例如 [52]中所示。因此，后文将采用有限差分作为正向算子 $F$。通过利用 PyTorch 的卷积网络基础设施实现差分模板，本文采用了一种基础但相对高效的实现。对于一个 500µ网格，完成全部 1200 个时间步的仿真运行约需 $252 \times 124$ 格点[1]秒实际时间。详情见附录 A。

引入不可训练前向算子后，理想情况下，与初边值问题相关的损失分量应非常小，即 $\mathcal{L}_\mathcal{N}$，$\mathcal{L}_\mathcal{B}$， $\mathcal{L}_\mathcal{I}$。由于建模误差和离散化误差，这些分量通常只能近似为零。它们的

[1]在 Nvidia A100-SXM4-40GB 上进行了测试

贡献在测量损失 $\mathcal{L}_{\mathcal{M}}$面前仍可忽略不计。因此，可以将它们从代价函数 $\mathcal{L}$中去除，使其现在仅由测量损失 $\mathcal{L}_{\mathcal{M}}$构成。

引入不可训练的正向算子会显著降低优化任务的复杂度。若采用时间步进方案作为正向算子 $F$，图3所示设计的缺点是灵敏度计算的开销更大 $\nabla_{\boldsymbol{\theta}}\mathcal{L}(F(\hat{\boldsymbol{\gamma}})) = \nabla_{\hat{\boldsymbol{\gamma}}}\mathcal{L}(F(\hat{\boldsymbol{\gamma}})) \cdot \nabla_{\boldsymbol{\theta}}\hat{\boldsymbol{\gamma}}(\boldsymbol{\theta})$，因为需要通过自动微分，使反向传播经过正向算子的每个时间步 $F$，从而计算 $\nabla_{\hat{\boldsymbol{\gamma}}}\mathcal{L}(F(\hat{\boldsymbol{\gamma}}))$。计算反演场相对于网络参数的梯度 $\nabla_{\boldsymbol{\theta}}\hat{\boldsymbol{\gamma}}(\boldsymbol{\theta})$ 的开销则保持不变。

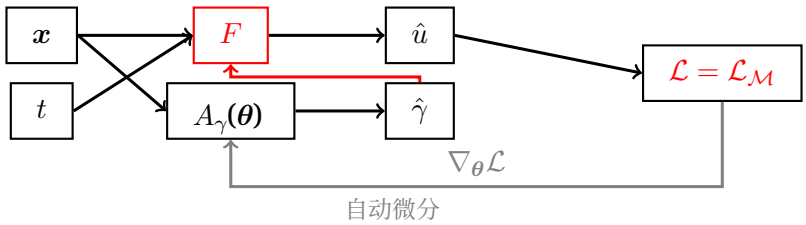



图3：采用不可训练前向算子的物理信息神经网络。图2 中原始物理信息神经网络的改动以红色标出。该方法不再学习正问题，而是使用不可训练的正向算子 $F$，并将 $\hat{\gamma}$ 作为额外输入。与此同时，这使代价函数 $\mathcal{L}$得到简化，因为 $\mathcal{L}_{\mathcal{N}}$， $\mathcal{L}_{\mathcal{B}}$， $\mathcal{L}_{\mathcal{I}}$由于$F$而近似满足。

## 2.2. 伴随方法

下一步是将指示函数 $\gamma$ 的神经网络试探函数替换为分段多项式试探函数。此处选择分段常数试探函数，并在一维情形下定义为 $\hat{\gamma}(x) = \sum_{i=1}^{N_c} c_i N_i(x)$，其中

$$N_i(x) = \begin{cases} 1 \text{ if } x_i \leq x < x_{i+1} \\ 0 \text{ else.} \end{cases} \tag{23}$$

对于二维和三维情形，额外维度的离散化处理方式相同。这一修改如图 4所示。系数$\boldsymbol{c} = \{c_i\}_{i=1}^{N_c}$仍使用 Adam 优化器更新，所用梯度通过自动微分获得。选择分段常数试探函数而非分段高阶多项式试探函数，是因为分段常数试探函数已能够表示从常数材料到空洞的跳变，正如第 3 节所示的所有示例。去掉神经网络后，Sigmoid激活函数也随之消失，因此必须以其他方式约束 $\hat{\gamma}$ 的取值范围。为此，采用额外的权重，即系数截断，将绝对值截断在一个接近零的小值 $\epsilon$ 与上界 1 或略大于 1 的值 [37]之间。

根据方程（23），关于系数的灵敏度

$$\nabla_{\boldsymbol{c}}\mathcal{L}(F(\hat{\boldsymbol{\gamma}})) = \nabla_{\hat{\boldsymbol{\gamma}}}\mathcal{L}(F(\hat{\boldsymbol{\gamma}})) \cdot \nabla_{\boldsymbol{c}}\hat{\boldsymbol{\gamma}}(\boldsymbol{c}) \tag{24}$$

可以通过自动微分或连续伴随方法计算，如图4所示。在连续伴随方法中，梯度在离散化之前进行解析推导，例如推导见 [37]。这一第三次修改将该方法转变为基于有限差分的经典全波形反演。由于这一修改，预计精度不会受到重大影响，因为梯度是相同的，除了伴随方法中有限差分离散化引入的数值误差。这些误差源于 Fr´echet 核

$$K_\gamma = \int_{\mathcal{T}} -\rho_0 u_t^\dagger u_t + \rho_0 c_0^2 \nabla u^\dagger \cdot \nabla u dt \tag{25}$$

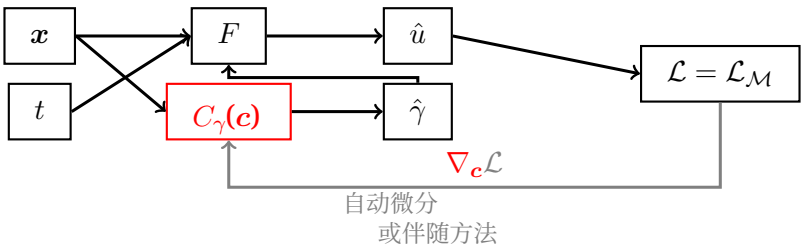



图 4：伴随方法。图 3 中先前修改过的物理信息神经网络的进一步修改以红色标出。神经网络试探函数被替换为一个由系数参数化的常数试探函数 $\boldsymbol{c}$。

用于计算梯度 $\nabla_{\hat{\gamma}}\mathcal{L}_\mathcal{M}$，通过在空间与时间域上对 $K_\gamma$ 进行积分（详见 [37] ）。为了计算方程（25），利用有限差分计算解场 $u$和伴随场 $u^\dagger$ 的时间梯度与空间梯度。请注意，相对于系数 $c_i$ 的灵敏度是在第二步中通过与相应的形函数 $N_i(x)$相乘计算得到的，该形函数通过对方程（23）相对于 $c_i$进行推导获得，即 $\nabla_{\boldsymbol{c}}\hat{}\boldsymbol{\gamma}(\boldsymbol{c})$。

事实上，伴随方法与自动微分之间的关键区别，在于梯度计算与状态方程离散化的先后顺序。在连续伴随方法中，首先通过解析方法推导梯度，然后按照上述方式对状态方程进行离散化。相反，在自动微分中，首先对微分方程进行离散化，然后将梯度沿每个时间步传播。因此，自动微分与离散伴随方法非常相似，并且如 [53], 所述，两者在本质上是相同的方法。像 [53, 54] 中那样，手动推导离散伴随方法中的灵敏度，或通过反向模式自动微分计算灵敏度，不会导致精度差异。因此，就精度而言，自动微分与连续伴随方法之间的差异仅受离散伴随方法与连续伴随方法之间差异的影响。

与精度不同，自动微分方法与连续伴随方法相比，计算成本有所不同。由于反向传播，自动微分会显著增加内存开销。此外，采用自动微分沿所有时间步进行反向传播的成本很高，正如常微分方程网络相关研究中所提到的 [55]。反向传播和连续伴随方法的理论复杂度相对于时间步数量而言，在这两种情况下都是 $O(n)$ [56]，而在实践中，两者的计算时间差异大约只有两倍。为节省内存，可以使用截断时间反向传播 [56, 57]，这通常用于循环神经网络。不过，第3节所研究的示例没有采用截断，因为发现这样会对指示函数的学习效果产生不利影响。

### 2.3. 混合方法

本节介绍一种混合方法，它结合了神经网络试探函数作为缩放函数 $\gamma$ 的优势，以及使用连续伴随方法，通过方程（9）计算损失泛函梯度（方程（25））时更高效的梯度计算。为此，重新将神经网络作为 $\gamma$的试探函数引入。类似于方程（24），代价函数的梯度可以通过链式法则计算，但现在必须针对神经网络参数 $\boldsymbol{\theta}$进行计算：

$$\nabla_{\boldsymbol{\theta}}\mathcal{L}(\hat{\boldsymbol{u}}(\hat{\boldsymbol{\gamma}}(\boldsymbol{\theta}))) = \nabla_{\hat{\boldsymbol{\gamma}}}\mathcal{L}(\hat{\boldsymbol{u}}(\hat{\boldsymbol{\gamma}})) \cdot \nabla_{\boldsymbol{\theta}}\hat{\boldsymbol{\gamma}}(\boldsymbol{\theta}) \tag{26}$$

代价函数相对于神经网络输出（即指标场）的灵敏度，$\nabla_{\hat{\boldsymbol{\gamma}}}\mathcal{L}$通过连续伴随方法方程（25）高效计算；而指标场相对于神经网络参数的灵敏度 $\nabla_{\boldsymbol{\theta}}\hat{}\boldsymbol{\gamma}$ 则通过神经网络中的反向传播 $A_\gamma(\boldsymbol{\theta})$ 获得。混合方法的计算时间与

在时间步数足够多时，伴随方法的计算时间相近。该方案如图5所示。近期在拓扑优化领域也提出了类似方案 [58]。

与上一节讨论的方案相比，混合方案除了节省内存外，还具有将神经网络与正演求解器分离的优点，因此不再严格要求正演求解器具备高效的 GPU 实现。相反，神经网络可以在 CPU 上提供指标预测 $\hat{\boldsymbol{\gamma}}$ 及其相对于参数 $\nabla_{\boldsymbol{\theta}}\hat{\boldsymbol{\gamma}}$ 的灵敏度，而正演求解器也可以独立地在 CPU 上运行。与正演求解器的开销相比，在 CPU 上而非 GPU 上评估和更新神经网络所增加的计算开销非常小。此外，正演求解器甚至可以是一个完全独立的黑箱求解器，用于提供波场 $\hat{u}$ 以及损失泛函 $\nabla_{\hat{\boldsymbol{\gamma}}}\mathcal{L}$的灵敏度。

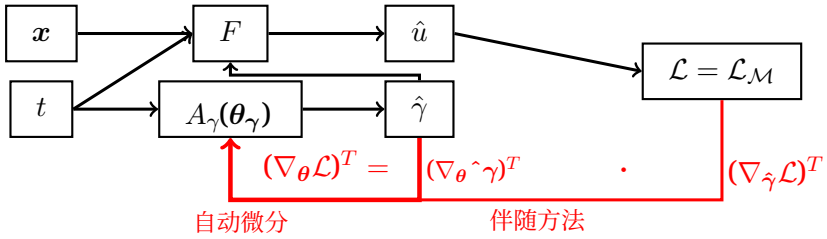



图5：混合方法。对先前修改过的 PINN（图3 ）所做的修改以红色标出。现在，梯度计算分为两部分。首先，使用伴随方法和方程（25）计算代价函数 $\mathcal{L}$ 相对于预测指标场 $\hat{\gamma}$ 的灵敏度，随后计算指标场 $\hat{\gamma}$ 相对于神经网络参数 $\boldsymbol{\theta}$ 的灵敏度，所用方法是训练神经网络时采用的自动微分。根据链式法则方程（26），即可得到所需的梯度 $\nabla_{\boldsymbol{\theta}}\mathcal{L}$ 。

## 3. 配置

本文考虑了两个复杂程度逐步增加的模型问题。首先，使用二维板来确定神经网络在反演过程中的优势与局限。之后，采用一个更复杂的三维情形，将所提出的混合方法与伴随方法进行比较。正向算子始终采用有限差分求解器。为避免逆问题犯罪 [59], 在部分域知识情形下，参考数据由边界协调有限元求解器生成（三维情形使用 Voxel-FEM）。然而，对于二维完整域知识情形，参考数据则采用有限单元法 [60] 生成。在所有研究中，均使用方程（8）所示的正弦突发信号，其周期数为 $n_c = 2$ ，频率为 $f_\psi = 500$ kHz，振幅为 $\psi_0 = 10^{12}\ \mathrm{kg(ms)^{-2}}$。在方程（7）的源施加过程中使用有限差分时，必须格外注意。狄拉克 delta 函数必须按比例缩放 $1/||d\boldsymbol{x}||_2^2$ ，以补偿网格间距 [61]。此外，两种情形采用相同的材料，其密度为 $\rho_0 = 2700\ \mathrm{kg/m^3}$ ，波速为 6000 $\mathrm{ms^{-1}}$。

### 3.1. 二维情况

二维情况深受以下内容的启发 [37], ：需要在一块长 100 mm 的 × 50mm 矩形板上检测一个直径为 5 mm 的圆形孔洞，如图7a所示。我们没有向机器学习算法提供有关孔洞的具体信息，这意味着整个材料分布 $\gamma(\boldsymbol{x})$都必须进行估计。虽然几何形状、材料和声源类型相同，但传感器和声源位置与 [37]略有不同。总共使用了 54 个传感器和 4 个声源，布置在顶边上，并进行了 4 次独立测量。具体配置如图6所示，其中展示了根据图1中的定义绘制的定义域左上边缘。该设置沿虚线标示的垂直中线对称

。传感器和声源位置的这一细微特殊性，是因为它们与有限差分节点和神经网络网格点对齐。然而，这并不是所述方法的限制，因为可以轻松通过例如使用凯泽窗sinc函数对声源进行插值 [61] 以及对传感器进行线性插值来规避这一问题。

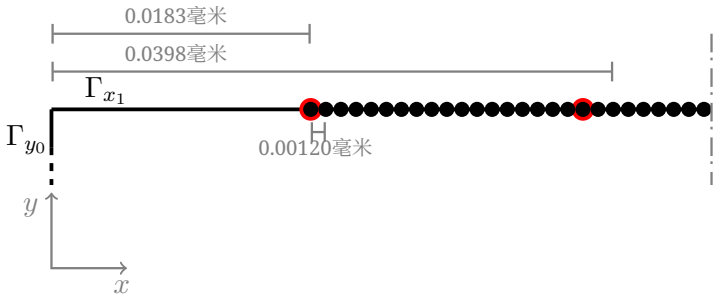


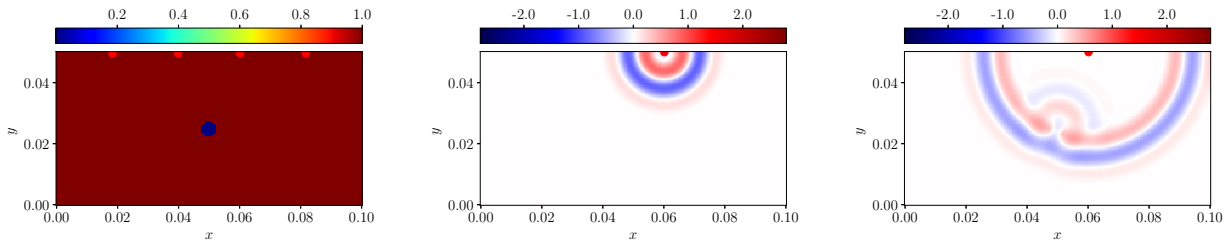

图6：对称的传感器（黑色圆点）和声源（红色圆点）配置 .

图中展示了一次正演模拟7，其中第三个源在 $t = 0$ 秒时发出一个脉冲，随后由于圆形空洞，中心处的波场受到扰动。用于反演的有限差分模拟使用 $252 \times 124$ 个网格点，每个维度有两个虚单元，指示函数下界为 $\epsilon = 10^{-5}$，时间步长为 $\Delta t = 3 \cdot 10^{-8}$ 秒$^{-1}$，共进行1200个时间步。

### 3.2. 三维情形

三维情形旨在展示，在伴随优化不再表现良好的情况下，神经网络可能具有的优势。为此，选取了来自一根 CT扫描钻芯 的两个长方体切片 [62] 进行研究，并展示于图8中。反演过程中，立方体的边长为 20 mm，每个维度离散为 92 个网格点；共考虑 1200 个时间步，时间步长为 $\Delta t = 1.8 \cdot 10^{-8}\,\mathrm{s}^{-1}$。同样，采用较低的指示函数下界 $\epsilon = 10^{-5}$，而伴随方法的上界则提高至 1。2，因为研究发现这样做有益。声源和传感器布置在立方体的背面，如图8右侧较深的阴影所示。四个声源距各边 5。27 mm，而传感器与该平面的网格点重合。

所选案例确保单次正演和伴随模拟可在 40 GB GPU内存的限制内完成。若要计算更大规模的三维问题，则必须采用高效的模拟并行化方案，可在 GPU集群上实现；或者在 CPU 上运行，并在神经网络的 GPU 与 CPU 之间进行高效通信。




(a) 由指示函数定义的材料 $\gamma(x, y)$

(b) 波场 $u(x,\ y,\ t)$，位于 $t = 0.00375\mu$秒。

(c) 波场 $u(x, y, t)$ 于 $t = 0.0075\mu$ 秒。

图 7：使用最左侧图中所示的底层材料分布和第三个声源计算得到的波场。当波到达缺陷时，可以看到波场受到扰动。

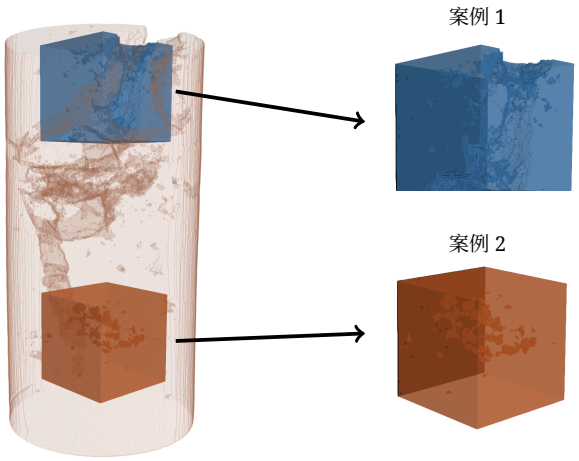



图 8：从 CT扫描钻芯中提取的两个案例 [62]。声源和传感器放置在立方体背面，即右侧较暗的表面上。

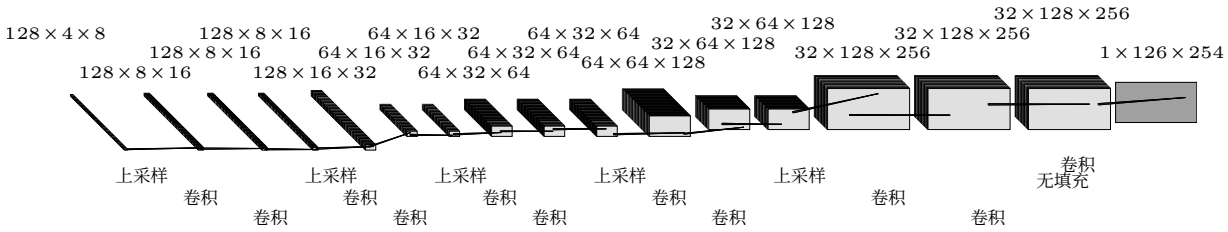
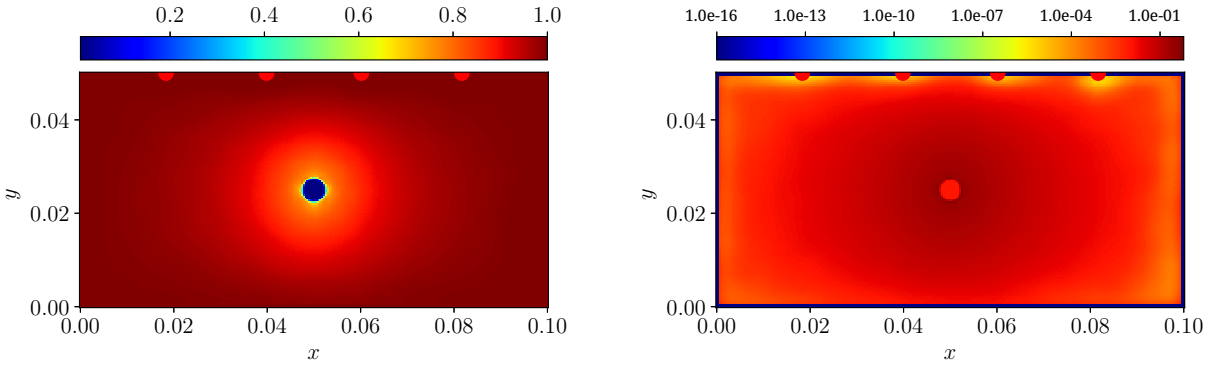



图 9：用于二维反演案例的卷积神经网络。它以大小为 $128 \times 4 \times 8$ 的随机输入，按照重复模式进行上采样和卷积，生成大小为 $\gamma$ 的预测 $\hat{} \ 1 \times 126 \times 254$。更多详情请参见附录 B。

### 3.3. 神经网络架构

作为指示函数 $\hat{}\gamma$ 的试探函数，该神经网络采用生成器结构，并使用受生成对抗网络 [63]启发的逐像素归一化。它将大小为 $128 \times 8 \times 4$ 的随机三维矩阵上采样，随后应用卷积滤波器。上采样和卷积重复五次，以获得指示函数 $\hat{}\gamma$所需的分辨率；二维情况所用的神经网络见图 9。三维情形采用类似的网络，使用三维卷积和大小为 $256 \times 3 \times 3 \times 3$的四维随机输入矩阵。为了提高指示体素数量与网络参数数量之比，三维情形使用了两倍数量的滤波器。有关这两种网络的更多细节，参见附录 B。上采样采用最近邻插值；与多项式插值相比，其效果更好，因为它能够保留表示空洞所需的不连续性。自适应激活函数因其加速收敛的特性 [64]，而被采用。具体而言，除最后一层外，所有层均使用 PReLU [65]；最后一层则使用自适应 Sigmoid，以确保输出处于范围 $[0, 1]$内。随后将输出重新缩放至范围 $[\epsilon, 1]$，以确保数值稳定性。为使比较更加直接，优化始终使用 Adam [44]，尽管 L-BFGS 所需的迭代次数更少。梯度裁剪 [66–68] 被发现对于确保早期收敛至关重要。若不进行梯度裁剪，训练会出现平台期，之后才会显著突破。此外，还采用了多项式衰减的学习率调度器（$b \cdot$ 训练轮次 $+ 1$）$^a$，其取值为 $a = -0.5$、$b = 0.2$。网络初始权重采用 Glorot 初始化 [69]。




(a) 预测。

(b) 绝对误差。

图10：使用具备完整域知识的物理信息神经网络训练100轮后材料分布的预测结果和误差。每个训练周期的平均耗时为0.51秒。

# 4. 数值结果

## 4.1. 二维情况

在后续章节中，我们将研究物理信息神经网络的各个组成部分，并将其与经典伴随优化进行比较，所用案例见第3.1节。所有模拟均在 Nvidia A100 SXM4 40GB GPU 上运行，并使用图9 和附录 B中介绍的同一卷积神经网络。

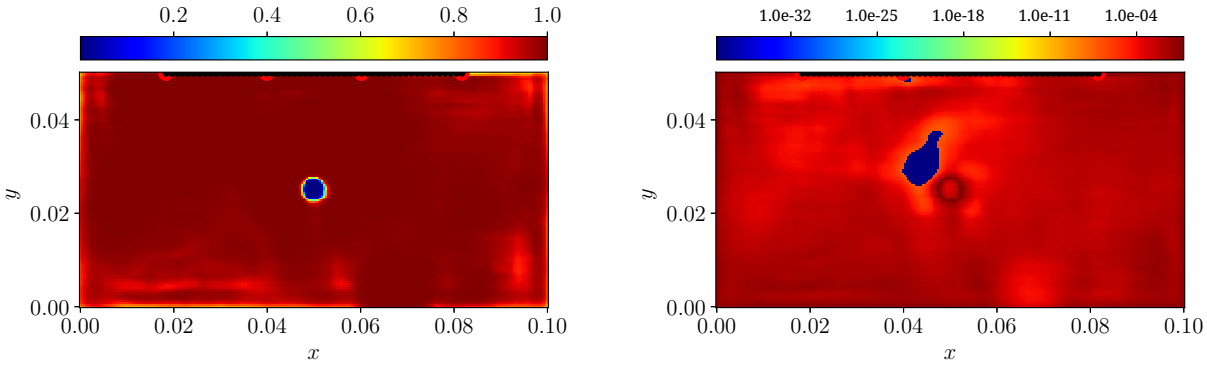
### 4.1.1. 物理信息神经网络

本节所呈现的结果对应于表 1中的物理信息神经网络案例，其方法在第 1.3 节中进行了描述。 [26] 所展示的各种反演任务耗时约 $400\,000 - 600\,000$ 个训练轮次，计算时间约为每个训练轮次 0.2 秒[2]。如此多的训练轮次说明了嵌套优化任务的复杂性：对于单个问题，需要同时学习正向解和反演解。应用多个声源时，问题会更加复杂，还必须学习一个能够考虑不同声源的正向算子。在 [26],这个问题通过源叠加得到规避 [70], ，因此只需学习一次正演模拟。

如果只需学习一个场（例如完整域知识条件下的反演，如 [25], 所示），优化任务的复杂性就会降低。为说明复杂度的降低以及物理信息神经网络在特定设置中的能力，我们使用带有额外传感器的物理信息神经网络求解第 3.1 节中提出的问题。传感器布置在整个定义域内，与有限差分网格点重合。由于测量值仅在离散点上给出，因此使用有限差分对导数项进行数值计算，从而数值评估微分方程（18）的残差。此外，在边界处指定指示函数，即将其设为 1，以确保边界处存在材料；这对于获得准确的反演结果至关重要。图 10 展示了经过 100 个训练轮次后的反演结果，图中清晰识别出了圆形空洞，且只有轻微伪影。必须在边界处指定密度$\rho(\boldsymbol{x})$，并使用相对较大的学习率 $\alpha_{\min} = 2 \cdot 10^{-3}$。此外，采用注意力机制的可学习惩罚权重也很有帮助，其学习率为 $\alpha_{\max} = 2 \cdot 10^{-2}$。

需要强调的是，从 $400\,000 - 600\,000$ 个训练轮次改进到100个训练轮次，是通过利用完整域知识简化物理问题所取得的结果，不应将其视为物理信息神经网络在部分域知识情况下所能实现效果的参考

[2]基于以下人员提供的 Python 脚本： [51] 并在 Nvidia A100-SXM4-40GB GPU 上进行了测试。




(a) 预测。 （b）绝对误差。

图 11：使用自动微分和神经网络试探函数训练 50 个训练轮次后材料分布的预测结果与误差。每个训练周期的平均耗时为 2.53 秒。

如第3.1节所述。相反， $400\,000 - 600\,000$ 个训练轮次应被视为物理信息神经网络处理类似反演任务能力的参考。

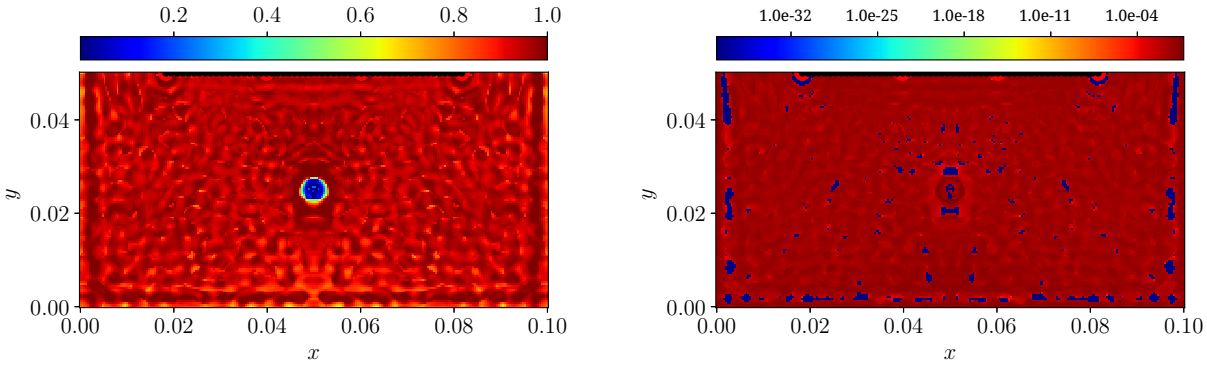
#### 4.1.2. 使用不可训练前向算子的物理信息神经网络

降低同时学习正向解和反演解的复杂度，另一种方法是采用表 1 中归类为使用不可训练前向算子的物理信息神经网络、并在第 2.1 节中介绍的不可训练前向算子。现在，用有限差分格式替代负责求取正向解的神经网络。仅经过 50 个训练轮次，即可得到图 11 所示的合理反演结果。因此，相较于具有完整域知识的物理信息神经网络，其按迭代次数衡量的收敛速度更快，但每个训练周期的耗时更长，为 2.53 秒。总体而言，考虑到在仅具备部分域知识的情况下， $400\,000 - 600\,000$ 个训练轮次中观察到的 [26] 结果，采用不可训练前向算子的物理信息神经网络表现更优。优化时采用了学习率 $\alpha_{\min} = 2{\cdot}10^{-3}$，并使用梯度裁剪。可学习惩罚权重略微改善了结果，但并非必不可少。

#### 4.1.3. 伴随方法

接下来，表 1 对伴随方法进行性能评估。第 2.2 节将进一步介绍相关细节，并重点说明试探函数和梯度计算的重要性。此处采用分段常数试探函数，而非使用自动微分进行梯度计算的神经网络。图 12所示，经过 50 个训练轮次后，预测结果清晰地识别出了空洞，且所用学习率略有提高，为 $\alpha_{\min} = 6 \cdot 10^{-2}$。然而，也观察到了额外的振荡伪影。这与 [37],中的观察结果一致，其中未知材料场采用线性形函数作为试探函数。不过，有趣的是，将 $\gamma$ 的试探函数改为神经网络后，得到的局部极小值截然不同且更优。具体而言，对于 $\gamma$，使用分段常数试探函数（如方程（23）所示）时，均方误差为 $3.2 \cdot 10^{-3}$；而使用神经网络（如方程（12）所示）时，均方误差降至 $9.5 \cdot 10^{-4}$。此外，在真实解为常数的区域，解的平滑性有所提高，同时圆形空洞周围的陡峭梯度得以保留，如图 11 所示，并在附录 C中进行了量化。值得注意的是，这种精度提升并未显著增加计算开销。

然而，必须指出，试探函数的变化还伴随着一个主要差异。常数试探函数的初始场在整个定义域内均被设为 1，而在神经网络情形下，如果不进行预训练，这是不可能实现的。通过预训练使神经网络学习一个值为 1 的常数场，并同时施加较强的正则化，并未发现有帮助。因此，神经网络从一个随机场开始，正如图 16 中随后展示的预测历史所示。然而，




(a) 预测。

（b）绝对误差。

图 12：使用自动微分和常数试探函数训练 50 轮次后的材料分布预测及误差。每个训练周期的平均耗时为 2.46 s。

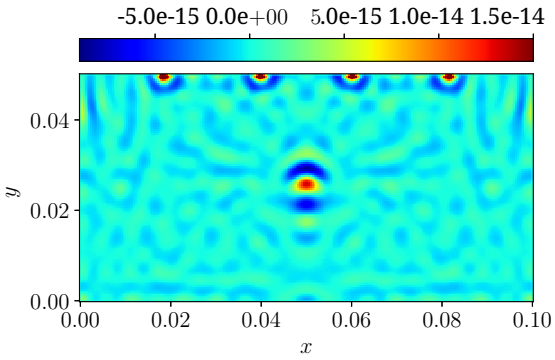
将随机场作为常数试探函数的初始化被发现会产生不利影响，这表现为无法成功收敛到有意义的反演结果。因此，神经网络试探函数的优势必然在于其非线性连接性和强大的过参数化能力：其参数数量为 526 252 个，而常数试探函数的参数数量为 $252 \times 124 = 31\,248$ 个。

需要注意的是，增加常数试探函数中的参数数量可能会带来益处，并使其与神经网络试探函数的比较更加公平。然而，使用有限差分时，增加参数数量（例如增加体素数量，或提高每个有限差分网格点上的多项式次数）并不会改善指标的离散化。在有限元法和单元法中可以采用这些方法，因为可以通过在每个单元中使用多个高斯点来纳入新增参数。因此，神经网络作为离散化工具的优势更加明显，因为它们提供了一种不受模拟方法限制的通用工具。

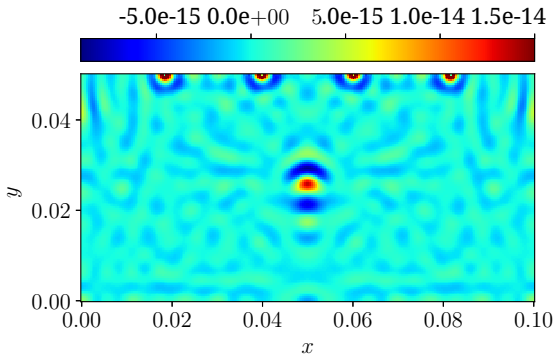
经过这项最终修改，[37]中介绍的经典伴随优化就几乎完成了，唯一细微的区别是使用自动微分进行梯度计算。不过，如前所述，从自动微分改用连续伴随方法来计算梯度，不太可能产生显著影响。为通过数值方式对此进行评估，分别使用两种方法计算常数试探函数下代价函数相对于反演场 $\nabla_{\boldsymbol{c}}\mathcal{L}$ 的初始梯度，并通过每次改变一个参数 $c_i$，将其与使用有限差分得到的参考解进行比较。只有边界处可观察到细微差异，如图13所示。这种差异可归因于 Fr´echet 核的数值计算（方程（25）），由于采用了中心有限差分格式，边界处产生了最大的数值误差。然而，相对于参考解的绝对误差仍可忽略不计。因此，预计图12中的学习预测只会存在细微差异，数值比较也证实了这一点。这些微小差异在迭代过程中逐渐累积，最终导致得到略有不同但反演质量相近的局部极小值。因此，无论使用自动微分还是伴随方法进行梯度计算，解的质量都不会有所改善。然而，使用伴随方法计算梯度 $\nabla_{\hat{\gamma}}\mathcal{L}_{\mathcal{M}}$ 可显著降低计算开销：对于与图*12*所示相近的模拟，每个训练周期的耗时可从 2.46 s 降至每周期 1.05 s。

### 4.1.4. 混合方法

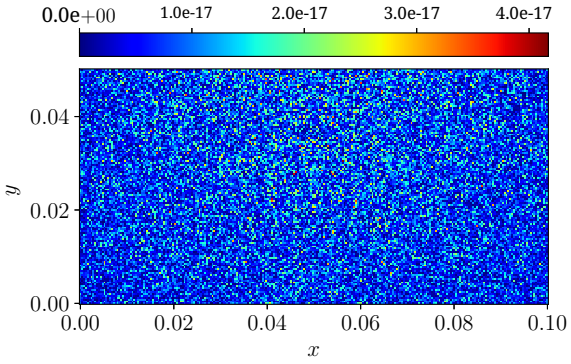
表1 中归类并在第2.3 节中详细介绍的混合方法，将神经网络试探函数的优势与伴随方法在梯度计算方面较低的计算开销结合起来 $\nabla_{\hat{\gamma}}\mathcal{L}_{\mathcal{M}}$。所得预测结果见图14，与图11相比，预测质量仅略有下降，同时每个训练轮次的计算时间仍为1。06秒。学习率 $\alpha_{\min} = 4 \cdot 10^{-3}$ 略有提高



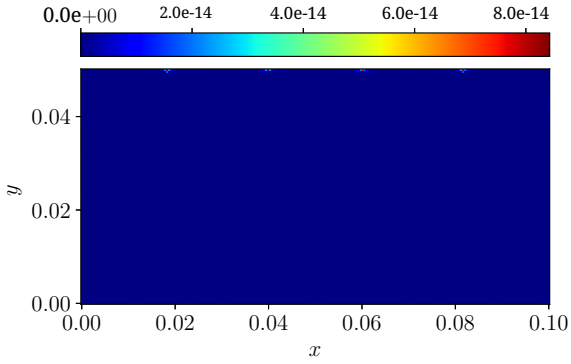

(a) 梯度，自动微分。




(b) 梯度，伴随方法。

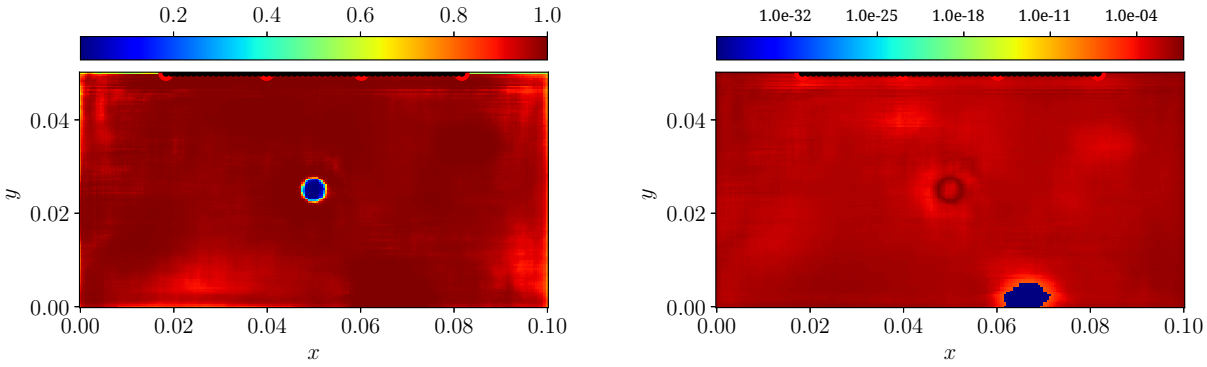



(c) 绝对误差，自动微分。




(d) 绝对误差，伴随方法。

图13：使用自动微分（a）和伴随方法（b）计算的常数试探函数初始梯度。（c）和（d）中的相应绝对误差使用通过有限差分获得的数值参考解计算。




（a）预测。　　（b）绝对误差。

图 14：使用自动微分与伴随方法以及神经网络试探函数的组合，经过 50 个训练轮次后材料分布的预测结果和误差。每个训练轮次的平均耗时为 1.06 秒。

有益。

#### 4.1.5. 比较

为总结所研究方法之间的比较，图 15 对训练历史进行了并列展示。尽管常数试探函数能够使代价函数实现最佳最小化，但均方误差在神经网络试探函数下最低，再次证实了神经网络试探函数的优势。此外，与使用有限差分的伴随方法相比，使用自动微分时能够持续获得更低的代价函数值和均方误差，从而使采用自动微分或伴随方法的影响更加清晰。鉴于计算方面的优势，精度的轻微损失是可以接受的。这一差异仅源于有限差分对 Fr´echet 核的计算，即公式（25）。对于有限元，Fr´echet 核可以通过对形函数进行解析微分而精确求得。

为了进一步了解训练行为，请参见图16所示的指标预测历史。图中并列展示了自动微分与神经网络试探函数、伴随方法与常数试探函数，以及混合方法在迭代特定时刻的快照，呈现整个定义域的情况。在这三种方法中，空洞均在 15 个训练轮次后基本得到重建。识别出空洞后，其余定义域仅有细微改善，其中混合方法的改善最明显，伴随方法的改善最弱。有趣的是，尽管神经网络方法一开始存在噪声，但在相近的计算开销下，这三种方法最终分别收敛至三个截然不同的局部极小值。

因此，本研究得出了两个重要结论。首先，对于单个反演问题，同时学习正向场和逆向场，比使用有限差分等不可训练前向算子会形成复杂得多的优化任务。若以迭代次数衡量，使用不可训练前向算子可带来约四个数量级的改进。其次，就重建场而言，以神经网络作为逆向场的试探函数，优于分段常数试探函数等经典方法。为再次凸显二者的差异，图17展示了通过自动微分优化后，神经网络试探函数和常数试探函数所得预测的曲面图。该曲面图呈现了两种解在平滑性上的差异。尽管神经网络解更平滑，但仍保留了锐利度（其量化方法见附录C）。该图表明存在正则化效应，下一节将通过三维情形进一步研究这一效应 4.2。

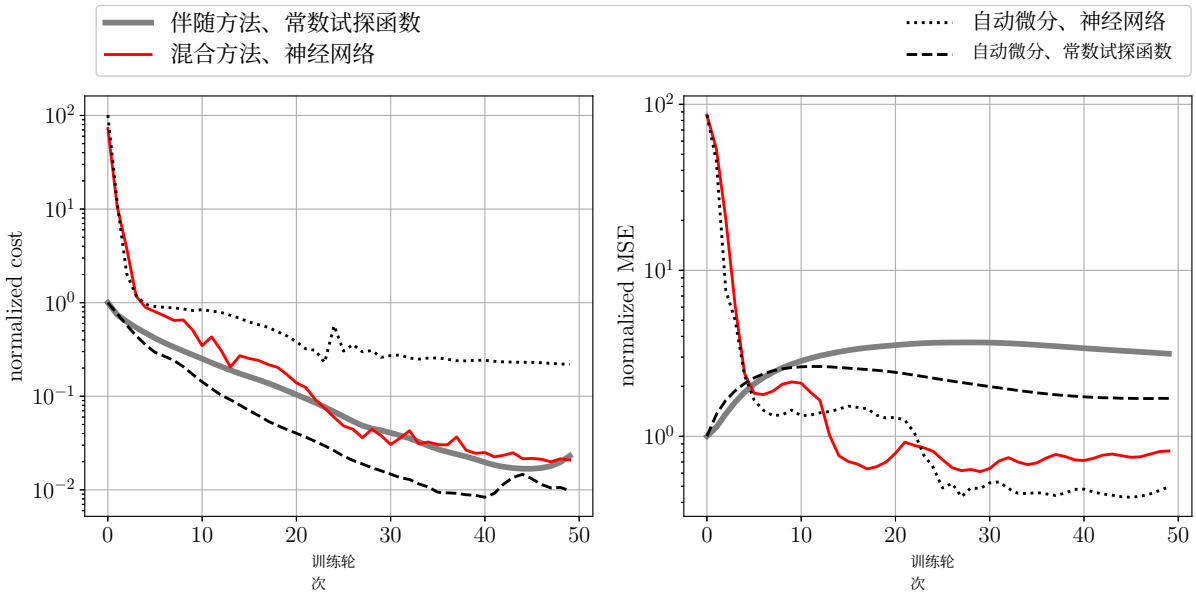



图 15：各方法训练历史的比较。左侧显示代价函数（由测量损失 $\mathcal{L}_{\mathcal{M}}$ 构成），右侧显示相对于真实材料分布 $\gamma(\boldsymbol{x})$ 的均方误差（MSE）。代价函数和均方误差均以整个定义域内指示函数等于 1 时的对应值进行归一化。

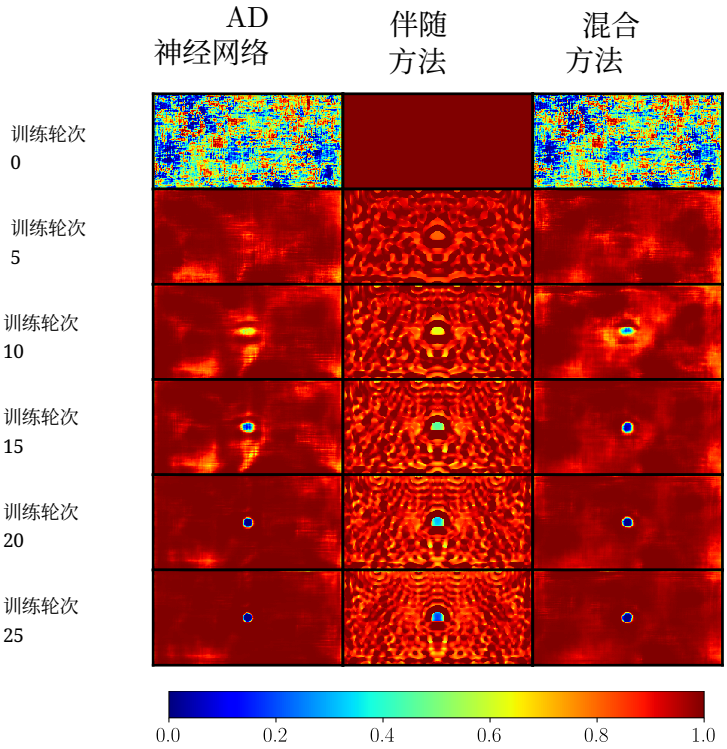
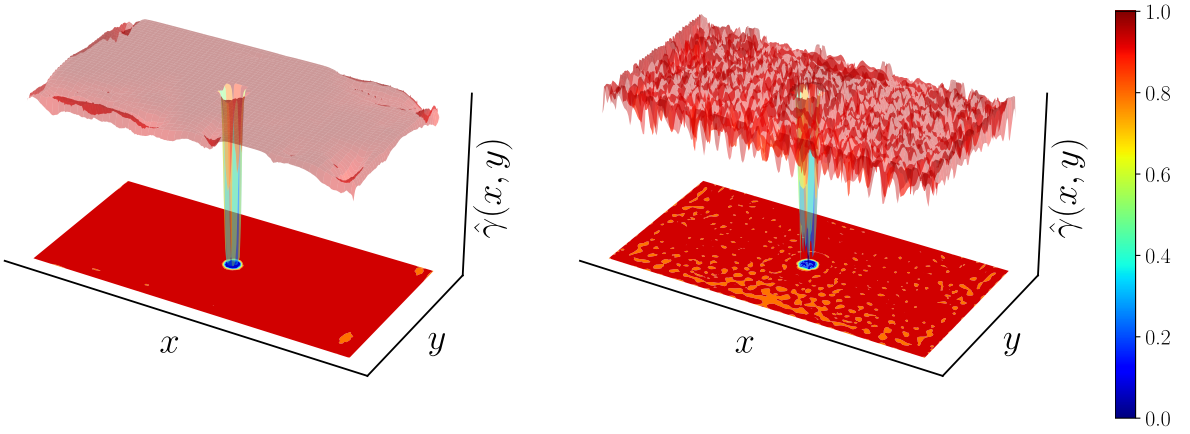



图16：从训练轮次0到训练轮次25的预测历史。从左到右依次展示了采用神经网络试探函数的自动微分、采用分段常数试探函数的伴随方法，以及采用神经网络试探函数的混合方法。




(a) 神经网络试探函数。　　(b) 常数试探函数。

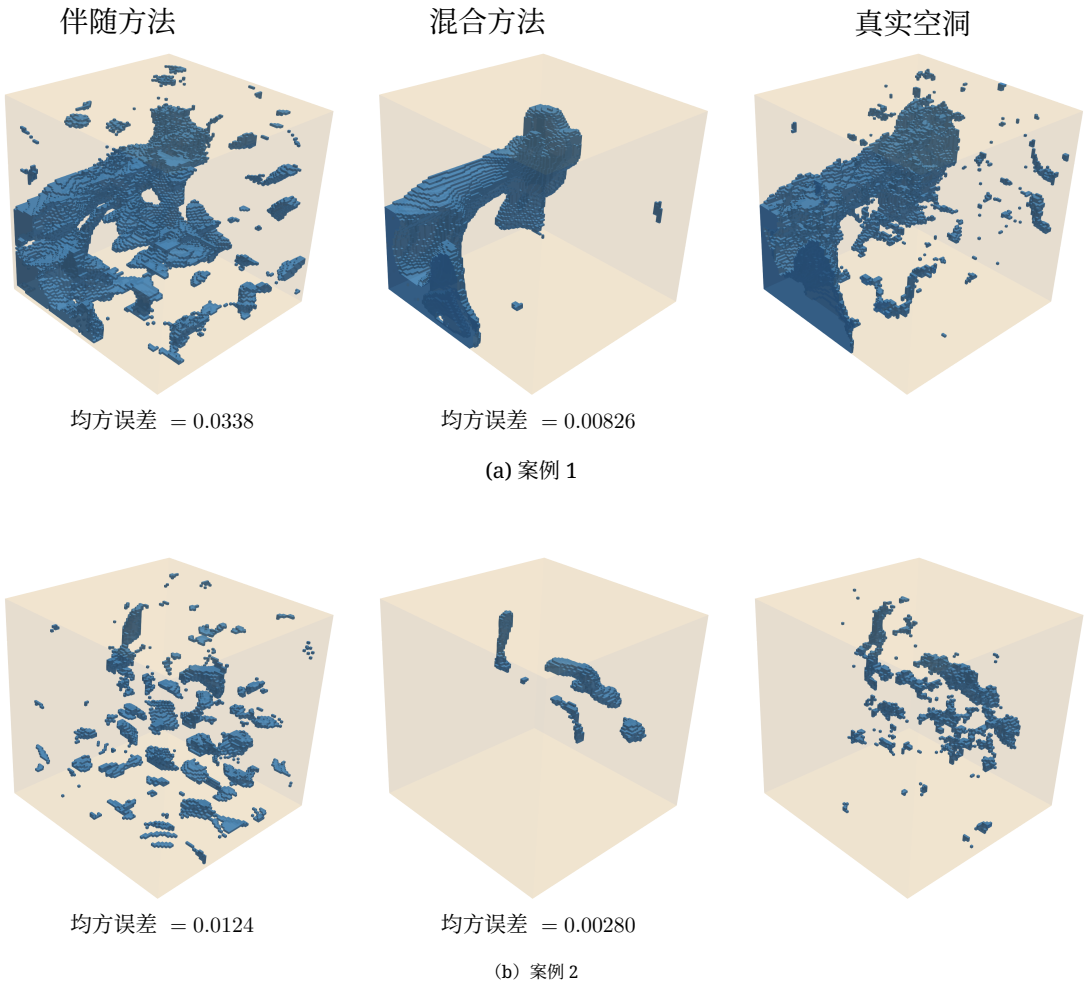
图17：使用自动微分，通过神经网络和常数试探函数进行训练后，经过50个训练轮次得到的预测材料分布曲面图。

### 4.2. 三维情形

鉴于使用神经网络作为试探函数所获得的反演结果质量较高，本文考虑三维情形。由于增加一个空间维度会带来额外的计算负担，因此仅比较效率最高且最有前景的方法，即混合方法和采用分段常数试探函数的伴随方法。两种方法的反演结果以识别出的空洞形式展示在图18 中，所针对的是图8中的两个CT扫描切片。再次证明，神经网络能够实现更优的重建，这可通过图18中标注的均方误差进行量化。图19展示了中心切片，以便了解解的整体分布和平滑性。由此可见，伴随方法的重建受到振荡伪影的严重影响，以至于无法再清晰识别空洞。混合方法的伪影少得多，能够以更高的精度恢复空洞。

我们将这种优越性归因于神经网络试探函数的不同特性：它不仅导致指标场过参数化，还具有内在的非线性。具体而言，神经网络使用了 6 306 764 个参数，而 $92^3 = 778\,688$经典离散化中的参数数量则为。值得注意的是，使用较少的神经网络参数时，重建效果会变差。将滤波器数量减半和减至四分之一后，参数数量分别仅为 1 577 452 和 394 748，重建质量随之下降。同时，将滤波器数量增加一倍，使参数数量达到 25 221 004，会导致质量略微下降，这是因为过参数化的正则化效应此时占据主导地位。图 20 中给出了网络三种变体相对于真实材料分布的均方误差（MSE），量化了这一观察结果。此外，还可以观察到平滑效应：参数数量增加表明其具有类似正则化的行为。因此，神经网络所实现的过参数化是提升反演结果的一个关键因素，但并非唯一因素。

神经网络与分段常数试探函数之间还有其他几个方面的差异。神经网络内部的非线性激活函数构成了一个重要区别。此外，神经网络充当全局函数，而常数试探函数则定义在局部区域上。此外，其他网络架构，例如全连接网络或图神经网络，可能表现出不同的行为。目前，除过参数化之外，尚不清楚哪些方面至关重要。所有这些方面都可以通过直接的实证方式进行研究，并且有待




(a) 案例 1

(b) 案例 2

图 18：伴随方法和混合方法在 100 个训练轮次后识别出的空洞，与真实空洞的比较。图示空洞通过对低于 0.5 的指示函数值进行阈值化提取。两种方法的每个训练周期的平均耗时均约为 29.1 秒。

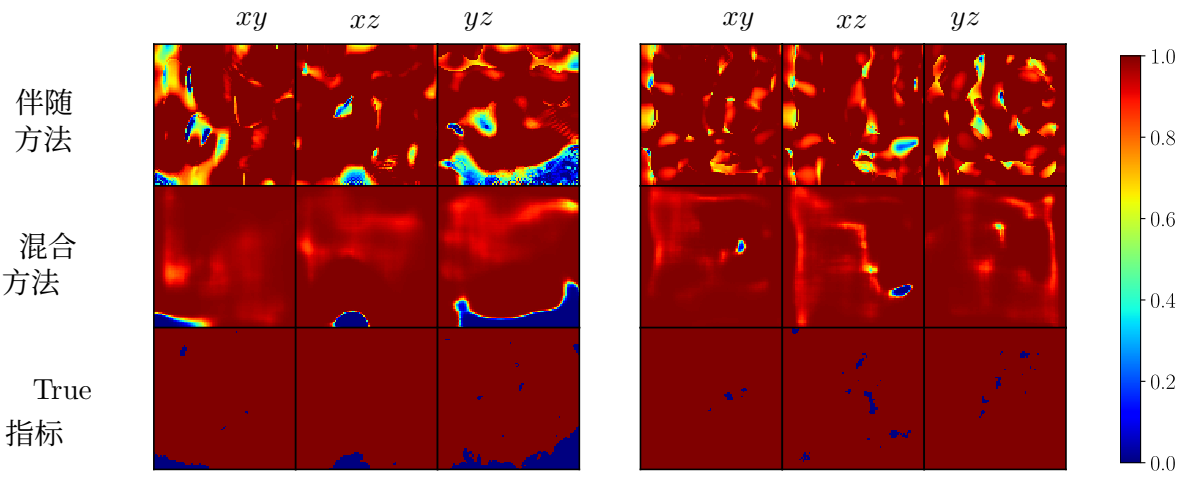
进一步研究。

## 5. 结论

本工作的目标是将物理信息神经网络与用于全波形反演的经典伴随优化进行比较，并通过逐步修改量化各方法的优势。首先，研究表明，通过使用两个神经网络进行嵌套优化来估计正向解并不理想。相反，建议采用经典正问题求解器，如伴随优化中所使用的求解器。其次，将神经网络作为反演场的试探函数非常有益，能够产生类似正则化的效果，同时不会损失锐度。第三，在比较自动微分和连续伴随方法时，灵敏度计算的精度不存在显著差异。然而，由于计算开销较低，在估计相对于反演场的代价函数梯度时，更推荐采用连续伴随方法。

基于这些认识，提出了混合方法：利用神经网络对逆向场进行离散化，并通过伴随方法计算关于逆向场的代价函数梯度，随后将其传递给反向传播算法，以更新神经网络参数。在实现相当计算时间的同时，逆重建结果得到了改善。

性　　神经网络作为逆向场离散化手段的性能不仅能够实现




(a) 案例 1　　(b) 案例 2

图19：使用伴随方法和混合方法训练 100 个训练轮次后的指示函数预测中心切片。切片平面标注于顶部。混合方法每个训练周期的平均耗时为 29.1 秒，伴随方法为 29.1 秒。

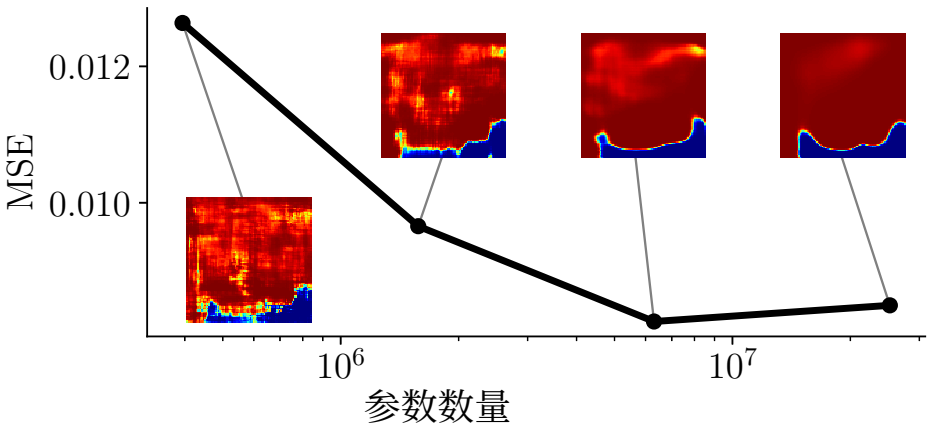



图20：案例 1 中不同参数数量对应的训练 100 个训练轮次后的均方误差。

进一步研究其他反问题，并深入探究其成功背后的根本原因，也同样值得关注。通过考察全局函数与局部函数之间的差异、不同的过参数化方式（即参数数量）、网络中的非线性、不同的网络架构，以及波场初始化的方式及其对反演的影响，或许能够获得启示。

## 致谢

作者衷心感谢巴伐利亚州科学与艺术部（StMWK）通过联合研究项目“巴伐利亚地热联盟”（GAB）提供资助，该项目资助了 Leon Herrmann；同时感谢 Georg Nemetschek Institut（GNI）通过“Deep-Monitor”项目提供资助。此外，作者衷心感谢德国研究联合会提供的资助，资助编号为 KO 4570/1-1 和 RA 624/29-1，该资助支持了 Tim B¨urchner。Felix Dietrich 谨此感谢德国研究联合会提供的资助，项目编号为 468830823。

## 声明

**利益冲突**作者声明不存在潜在利益冲突。

## 数据可用性

我们提供了一个 PyTorch [7]实现，涵盖了 [71]中的所有方法。

# A. 深度学习框架中的高效有限差分实现

以二维情况为例展示的显式有限差分格式，

$$
\begin{aligned}
u_{i,j}^{n+1} = & -u_{i,j}^{n-1} + 2u_{i,j}^{n} \\
& + \frac{2}{\gamma_{i,j}} \left( \frac{c_0 \Delta t}{\Delta x} \right)^2 \left( \left( \frac{1}{\gamma_{i,j}} + \frac{1}{\gamma_{i+1,j}} \right)^{-1} (u_{i+1,j}^{n} - u_{i,j}^{n}) - \left( \frac{1}{\gamma_{i-1,j}} + \frac{1}{\gamma_{i,j}} \right)^{-1} (u_{i,j}^{n} - u_{i-1,j}^{n}) \right) \\
& + \frac{2}{\gamma_{i,j}} \left( \frac{c_0 \Delta t}{\Delta y} \right)^2 \left( \left( \frac{1}{\gamma_{i,j}} + \frac{1}{\gamma_{i,j+1}} \right)^{-1} (u_{i,j+1}^{n} - u_{i,j}^{n}) - \left( \frac{1}{\gamma_{i,j-1}} + \frac{1}{\gamma_{i,j}} \right)^{-1} (u_{i,j}^{n} - u_{i,j-1}^{n}) \right) \\
& + \frac{\Delta t^2}{\rho_0 \gamma_{i,j}} f_{i,j}^{n}
\end{aligned}
$$

在 PyTorch 中实现 [7] 使用卷积核 $\boldsymbol{K}$，从而充分利用 PyTorch 的 GPU 能力。

$$
\begin{aligned}
\boldsymbol{u}^{n} = & -\boldsymbol{u}^{n} + 2\boldsymbol{u}^{n-1} \\
& + \frac{2}{\boldsymbol{\gamma}} \Bigg( \frac{1}{\frac{1}{\boldsymbol{\gamma}} * \boldsymbol{K}_{\gamma_{x_0}}} (\boldsymbol{u} * \boldsymbol{K}_{u_{x_0}}) - \frac{1}{\frac{1}{\boldsymbol{\gamma}} * \boldsymbol{K}_{\gamma_{x_1}}} (\boldsymbol{u} * \boldsymbol{K}_{u_{x_1}}) \\
& \quad + \frac{1}{\frac{1}{\boldsymbol{\gamma}} * \boldsymbol{K}_{\gamma_{y_0}}} (\boldsymbol{u} * \boldsymbol{K}_{u_{y_0}}) - \frac{1}{\frac{1}{\boldsymbol{\gamma}} * \boldsymbol{K}_{\gamma_{y_1}}} (\boldsymbol{u} * \boldsymbol{K}_{u_{y_1}}) \\
& \quad + \frac{\Delta t^2}{2\rho_0} \boldsymbol{f} \Bigg)
\end{aligned}
$$

相应的卷积核定义如下

$$
\boldsymbol{K}_{\gamma_{x_0}} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{pmatrix}, \boldsymbol{K}_{u_{x_0}} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & -\left(\frac{c_0 \Delta t}{\Delta x}\right)^2 & \left(\frac{c_0 \Delta t}{\Delta x}\right)^2 \\ 0 & 0 & 0 \end{pmatrix},
$$

$$
\boldsymbol{K}_{\gamma_{x_1}} = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \boldsymbol{K}_{u_{x_1}} = \begin{pmatrix} 0 & 0 & 0 \\ -\left(\frac{c_0 \Delta t}{\Delta x}\right)^2 & \left(\frac{c_0 \Delta t}{\Delta x}\right)^2 & 0 \\ 0 & 0 & 0 \end{pmatrix},
$$

$$
\boldsymbol{K}_{\gamma_{y_0}} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{pmatrix}, \boldsymbol{K}_{u_{y_0}} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & -\left(\frac{c_0 \Delta t}{\Delta y}\right)^2 & 0 \\ 0 & \left(\frac{c_0 \Delta t}{\Delta y}\right)^2 & 0 \end{pmatrix},
$$

$$
\boldsymbol{K}_{\gamma_{y_1}} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \boldsymbol{K}_{u_{y_1}} = \begin{pmatrix} 0 & -\left(\frac{c_0 \Delta t}{\Delta y}\right)^2 & 0 \\ 0 & \left(\frac{c_0 \Delta t}{\Delta y}\right)^2 & 0 \\ 0 & 0 & 0 \end{pmatrix}
$$

齐次诺伊曼边界条件通过虚单元（例如参见 [19]）强制实施，并通过使用池化层以及根据虚单元进行手动调整来实现。

# B. 网络架构

图 9 所示网络的定量总结见表3 （二维情形）和表4 （三维情形）。对于不同的方法和情形，网络超参数的选择略有不同。感兴趣的读者可参阅以下地址提供的代码 [71]。

表3：用于预测指示器 $\gamma$ 的二维卷积神经网络架构。参数总数为 526 252。

| **层** | **层后的形状** | **可学习参数** |
|---|---|---|
| 输入 | $128 \times 8 \times 4$ | 0 |
| 上采样 | $128 \times 16 \times 8$ | 0 |
| 二维卷积 & PReLU | $128 \times 16 \times 8$ | $147\,584 + 1$ |
| 二维卷积 & PReLU | $128 \times 16 \times 8$ | $147\,584 + 1$ |
| 上采样 | $128 \times 32 \times 16$ | 0 |
| 二维卷积 & PReLU | $64 \times 32 \times 16$ | $73\,792 + 1$ |
| 二维卷积 & PReLU | $64 \times 32 \times 16$ | $36\,928 + 1$ |
| 上采样 | $64 \times 64 \times 32$ | 0 |
| 二维卷积 & PReLU | $64 \times 64 \times 32$ | $36\,928 + 1$ |
| 二维卷积 & PReLU | $64 \times 64 \times 32$ | $36\,928 + 1$ |
| 上采样 | $64 \times 128 \times 64$ | 0 |
| 二维卷积 & PReLU | $32 \times 128 \times 64$ | $18\,464 + 1$ |
| 二维卷积 & PReLU | $32 \times 128 \times 64$ | $9\,248 + 1$ |
| 上采样 | $32 \times 256 \times 128$ | 0 |
| 二维卷积 & PReLU | $32 \times 256 \times 128$ | $9\,248 + 1$ |
| 二维卷积 & PReLU | $32 \times 256 \times 128$ | $9\,248 + 1$ |
| 无填充的二维卷积 & 自适应 Sigmoid | $1 \times 254 \times 126$ | $289 + 1$ |

表4：用于预测指标 $\gamma$ 的三维卷积神经网络架构。每层的滤波器数量是二维卷积神经网络的两倍。参数总数为 6 306 764。

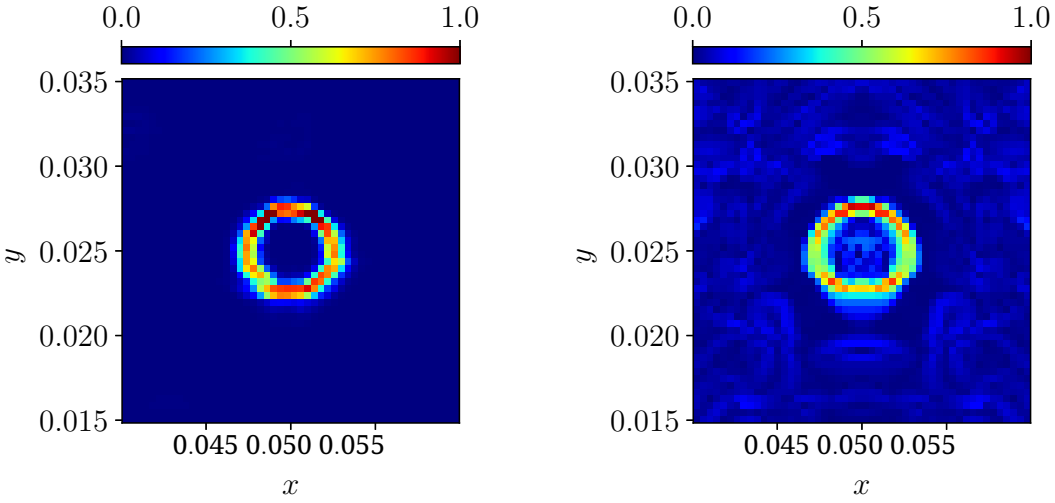
| **层** | **层后的形状** | **可学习参数** |
|---|---|---|
| 输入 | $256 \times 3 \times 3 \times 3$ | 0 |
| 上采样 | $256 \times 6 \times 6 \times 6$ | 0 |
| 三维卷积 & PReLU | $256 \times 6 \times 6 \times 6$ | $1\,769\,728 + 1$ |
| 三维卷积 & PReLU | $256 \times 6 \times 6 \times 6$ | $1\,769\,728 + 1$ |
| 上采样 | $256 \times 12 \times 12 \times 12$ | 0 |
| 三维卷积 & PReLU | $128 \times 12 \times 12 \times 12$ | $884\,864 + 1$ |
| 三维卷积 & PReLU | $128 \times 12 \times 12 \times 12$ | $442\,496 + 1$ |
| 上采样 | $128 \times 24 \times 24 \times 24$ | 0 |
| 三维卷积 & PReLU | $128 \times 24 \times 24 \times 24$ | $442\,496 + 1$ |
| 三维卷积 & PReLU | $128 \times 24 \times 24 \times 24$ | $442\,496 + 1$ |
| 上采样 | $128 \times 48 \times 48 \times 48$ | 0 |
| 三维卷积 & PReLU | $64 \times 48 \times 48 \times 48$ | $221\,248 + 1$ |
| 三维卷积 & PReLU | $64 \times 48 \times 48 \times 48$ | $110\,656 + 1$ |
| 上采样 | $64 \times 96 \times 96 \times 96$ | 0 |
| 三维卷积 & PReLU | $64 \times 96 \times 96 \times 96$ | $110\,656 + 1$ |
| 三维卷积 & PReLU | $64 \times 96 \times 96 \times 96$ | $110\,656 + 1$ |
| 无填充的三维卷积和自适应 Sigmoid | $1 \times 94 \times 94 \times 94$ | $1\,729 + 1$ |

## C. 锐度量化

受 Sobel滤波器（常用于图像处理）的启发，可通过恢复材料空间梯度的 $L_2$-范数来量化所识别材料分布的锐度[72]:

$$||\nabla\hat{\gamma}(\boldsymbol{x})||_2 = \sqrt{\nabla_x\hat{\gamma}(\boldsymbol{x}) + \nabla_y\hat{\gamma}(\boldsymbol{x})} \tag{27}$$

更大的范数表示更锐利的边缘。为突出显示与常数试探函数相比，神经网络试探函数在锐利度恢复方面的改进，图27式中圆形空洞周围的范数如图21 所示，针对二维情况。在圆形边界处，神经网络情形下的范数大多更大，显示出更锐利的反演。通过考察非零值的均值可对其进行量化；这些非零值是通过选取范数大于正阈值的点而提取的，目的是排除常数试探函数解中存在的噪声。对于任何阈值，神经网络情形下的均值都更大，证实了恢复材料分布整体上更为锐利。




(a) 图 11 中的神经网络试探函数11。

(b) 图 12 中的常数试探函数12。

图21：恢复材料分布 $||\nabla\hat{}\gamma(\boldsymbol{x})||_2$ 的空间梯度的 $L_2$ 范数。

# 参考文献


[1] I. Lagaris、A. Likas 和 D. Fotiadis，“用于求解常微分方程和偏微分方程的人工神经网络”，*IEEE Transactions on Neural Networks*，第 9 卷，第 987–1000 页，1998年9月。[2] D. C. Psichogios 和 L. H. Ungar，“一种用于过程建模的混合神经网络—第一性原理方法”，*AIChE Journal*，第 38 卷，第 1499–1511 页，1992年10月。[3] M. Raissi、P. Perdikaris 和 G. Karniadakis，“物理信息神经网络：用于求解涉及非线性偏微分方程的正问题和反问题的深度学习框架”，*Journal of Computational Physics* 378，686 707，2019，第 卷，第 – 页，2月。[4] S. Cuomo、V. S. Di Cola、F. Giampaolo、G. Rozza、M. Raissi 和 F. Piccialli，“通过物理信息神经网络实现科学机器学习：我们所处的位置及下一步方向”，*Journal of Scientific Computing*，第 92 卷，第 88 页，2022年7月。[5] G. E. Karniadakis、I. G. Kevrekidis、L. Lu、P. Perdikaris、S. Wang 和 L. Yang，“物理信息机器学习”，*Nature Reviews Physics*，第 3 卷，第 422–440 页，2021年6月。[6] S. Kollmannsberger、D. D’Angella、M. Jokeit 和 L. Herrmann，*Physics-Informed Neural Networks*，第 977 卷，第 55–84 页。沙姆：施普林格国际出版公司，2021年。[7] A. Paszke、S. Gross、F. Massa、A. Lerer、J. Bradbury、G. Chanan、T. Killeen、Z. Lin、N. Gimelshein、L. Antiga、A. Desmaison、A. K¨opf、E. Yang、Z. DeVito、M. Raison、A. Tejani、S. Chilamkurthy、B. Steiner、L. Fang、J. Bai 和 S. Chintala，“PyTorch：命令式风格的高性能深度学习库”，2019年12月。arXiv:1912.01703 [cs, stat]。[8] M. Abadi、A. Agarwal、P. Barham、E. Brevdo、Z. Chen、C. Citro、G. S. Corrado、A. Davis、J. Dean、M. Devin、S. Ghemawat、I. Goodfellow、A. Harp、G. Irving、M. Isard、Y. Jia、R. Jozefowicz、L. Kaiser、M. Kudlur、J. Levenberg、D. Man´e、R. Monga、S. Moore、D. Murray、C. Olah、M. Schuster、J. Shlens、B. Steiner、I. Sutskever、K. Talwar、P. Tucker、V. Vanhoucke、V. Vasudevan、F. Vi´egas、O. Vinyals、P. Warden、M. Wattenberg、M. Wicke、Y. Yu 和 X. Zheng，“TensorFlow：异构系统上的大规模机器学习”，2015年。软件可从 tensorflow.org 获取。[9] N. Thuerey、K. Weißenow、L. Prantl 和 X. Hu，“用于翼型流动雷诺平均纳维–斯托克斯模拟的深度学习方法”，*AIAA Journal*，第 58 卷，第 25–36 页，2020年1月。[10] Y. Zhu、N. Zabaras、P.-S. Koutsourelakis 和 P. Perdikaris，“无标注数据的高维代理建模和不确定性量化的物理约束深度学习”，*Journal of Computational Physics* 394，56 81，2019，第 卷，第 – 页，10月。[11] M. Lino、C. Cantwell、A. A. Bharath 和 S. Fotiadis，“利用多尺度图神经网络模拟连续介质力学”，2021年6月。arXiv:2106.04900 [physics]。[12] A. Sanchez-Gonzalez、J. Godwin、T. Pfaff、R. Ying、J. Leskovec 和 P. W. Battaglia，“利用图网络学习模拟复杂物理过程”，2020年9月。arXiv:2002.09405 [physics, stat]。[13] T. Pfaff、M. Fortunato、A. Sanchez-Gonzalez 和 P. W. Battaglia，“利用图网络学习基于网格的模拟”，2021年6月。arXiv:2010.03409 [cs]。[14] S. Bhatnagar、Y. Afshar、S. Pan、K. Duraisamy 和 S. Kaushik，“使用卷积神经网络预测气动流场”，*Computational Mechanics*，第 64 卷，第 525–545 页，2019年8月。[15] L. Lu、P. Jin、G. Pang、Z. Zhang 和 G. E. Karniadakis，“基于算子通用逼近定理通过 DeepONet 学习非线性算子”，*Nature Machine Intelligence*，第 3 卷，第 218–229 页，2021年3月。[16] S. Markidis，“旧与新：物理信息深度学习能否取代传统线性求解器？”，*Frontiers in Big Data*，第 4 卷，2021年。[17] R. Leiteritz 和 D. Pfl¨uger，“如何避免物理信息神经网络中的平凡解”，2021年12月。arXiv:2112.05620 [cs, stat]。[18] T. J. R. Hughes，*The finite element method: linear static and dynamic finite element analysis*。纽约州米尼奥拉：多佛出版社，2000年。[19] H. P. Langtangen，*Finite difference computing with PDEs: a modern software approach*。计算科学与工程教材丛书第 16 卷，瑞士沙姆：Springer Open，2017年。[20] J. Han、A. Jentzen 和 W. E，“使用深度学习求解高维偏微分方程”，*Proceedings of the National Academy of Sciences*，第 115 卷，第 8505–8510 页，2018年8月。[21] J. Sirignano 和 K. Spiliopoulos，“DGM：一种用于求解偏微分方程的深度学习算法”，*Journal of Computational Physics*，第 375 卷，第 1339–1364 页，2018年12月。[22] S. Goswami、A. Bora、Y. Yu 和 G. E. Karniadakis，“物理信息深度神经算子网络”，2022年7月。arXiv:2207.05748 [cs, math]。[23] J. Oldenburg、F. Borowski、A. ¨Oner、K.-P. Schmitz 和 M. Stiehm，“用于求解纳维–斯托克斯方程的几何感知物理信息神经网络代理模型（GAPINN）”，*Advanced Modeling and Simulation in Engineering Sciences* 9，8，2022，第 卷，第 页，6月。[24] J. C. Wong、C. Ooi、P.-H. Chiu 和 M. H. Dao，“利用物理信息神经网络改进流体动力学代理建模”，2021年5月。arXiv:2105.01838 [physics]。[25] K. Shukla、P. C. Di Leoni、J. Blackshire、D. Sparkman 和 G. E. Karniadakis，“用于表面开口裂纹超声无损定量的物理信息神经网络”，*Journal of Nondestructive Evaluation* 39，61，2020，第 卷，第 页，9月。[26] M. Rasht-Behesht、C. Huber、K. Shukla 和 G. E. Karniadakis，“用于波传播和全波形反演的物理信息神经网络（PINNs）”，*Journal of Geophysical Research: Solid Earth*，第 127 卷，2022年5月。[27] S. Cai、Z. Wang、F. Fuest、Y. J. Jeon、C. Gray 和 G. E. Karniadakis，“浓缩咖啡杯周围的流动：通过物理信息神经网络从断层背景导向纹影中推断三维速度和压力场”

网络 ," *Journal of Fluid Mechanics*, vol. 915, May 2021.
[28] S. Wang 和 P. Perdikaris, 《自由边界与 Stefan 问题的深度学习》, *Journal of Computational Physics* 428, 109914, 2021 , 第 p 卷, 2020
年3月。[29] Z. Mao、A. D. Jagtap 和 G. E. Karniadakis, 《高速流动的物理信息神经网络》, *Computer Methods in Applied Mechanics and Engineering* 360, 112789, 2020 , 第 p 卷, 2020
年3月。[30] A. D. Jagtap、E. Kharazmi 和 G. E. Karniadakis, 《离散域上的守恒物理信息神经网络：守恒律在正问题与反问题中的应用》, *Computer Methods in Applied Mechanics and Engineering* 365, 113028, 2020 , 第
p 卷, 6月。[31] A. D. Jagtap、Z. Mao、N. Adams 和 G. E. Karniadakis, 《超声速流动反问题的物理信息神经网络》, *Journal of Computational Physics*, 第466卷, 第111402页, 2022年10月。arXiv:2202.11821 [cs, math].[32] Y. Chen、L. Lu、G. E. Karniadakis 和 L. Dal Negro, 《纳米光学与超材料反问题的物理信息神经网络》, *Optics Express*, 第28卷, 第11618页, 2020年4月。[33] D. Givoli, 《反问题伴随方法教程》, *Computer Methods in Applied Mechanics and Engineering* 380, 113810, 2021 , 第 p 卷, 7月。
[34] R.-E. Plessix, 《利用伴随状态法计算具有地球物理应用的泛函梯度的综述》, *Geophysical Journal International*, 第167卷, 第495–503页, 2006年11月。[35] A. Fichtner 和 F. Bleibinhaus, *Full seismic waveform modelling and inversion*。《地球物理与环境力学和数学进展》, 柏林、海德堡：Springer, 2011年。[36] A. Sayag 和 D. Givoli, 《使用时变伴随法进行散射体形状识别》, *Computer Methods in Applied Mechanics and Engineering* 394, 114923, 2022 , 第 p 卷, 5月。
[37] T. B¨urchner、P. Kopp、S. Kollmannsberger 和 E. Rank, 《用于全波形反演的浸入边界参数化》, *Computer Methods in Applied Mechanics and Engineering*, 第406卷, 第115893页, 2023年3月。[38] A. G. Baydin、B. A. Pearlmutter、A. A. Radul 和 J. M. Siskind, 《机器学习中的自动微分：综述》, 2018年2月。arXiv:1502.05767 [cs, stat].[39] N. Geneva 和 N. Zabaras, 《利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》, *Journal of Computational Physics*, 第403卷, 第109056页, 2020年2月。arXiv: 1906.05747.[40] F. Wang、M. Jiang、C. Qian、S. Yang、C. Li、H. Zhang、X. Wang 和 X. Tang, 《用于图像分类的残差注意力网络》, 载于 *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 第6450–6458页, 2017年7月。ISSN: 1063-6919。[41] S. Zhang、J. Yang 和 B. Schiele, 《通过卷积神经网络中的引导注意力进行遮挡行人检测》, 载于 *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 第6995–7003页, 2018年6月。ISSN: 2575-7075。[42] Y. Nandwani、A. Pathak、Mausam 和 P. Singla, 《带约束深度学习的原始-对偶形式》, 载于 *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, 第1091号, 第12179–12190页, 美国纽约州雷德胡克：Curran Associates Inc., 2019年12月。[43] L. McClenny 和 U. Braga-Neto, 《使用软注意力机制的自适应物理信息神经网络》, 2022年4月。arXiv:2009.04544 [cs, stat].[44] D. P. Kingma 和 J. Ba, 《Adam：一种随机优化方法》, 2017年1月。arXiv:1412.6980 [cs].[45] D. C. Liu 和 J. Nocedal, 《关于大规模优化的有限内存BFGS方法》, *Mathematical Programming* 45, 503 528, 1989 , 第 pp 卷, 第–
页, 8月。[46] E. Samaniego、C. Anitescu、S. Goswami、V. Nguyen-Thanh、H. Guo、K. Hamdia、X. Zhuang 和 T. Rabczuk, 《通过机器学习求解计算力学中偏微分方程的能量方法：概念、实现与应用》, *Computer Methods in Applied Mechanics and Engineering* 362, 112790, 2020 , 第 p 卷, 4月。
[47] B. Moseley、A. Markham 和 T. Nissen-Meyer, 《利用物理信息深度学习求解波动方程》, 2020年6月。arXiv:2006.11894 [physics].[48] C. Song、T. Alkhalifah 和 U. B. Waheed, 《使用物理信息神经网络求解频域声学VTI波动方程》, *Geophysical Journal International*, 第225卷, 第846–859页, 2021年2月。[49] C. Song、T. Alkhalifah 和 U. b. Waheed, 《使用物理信息神经网络求解声学VTI波动方程》, *Geophysical Journal International*, 第225卷, 第846–859页, 2021年2月。arXiv:2008.01865 [physics].[50] S. Karimpouli 和 P. Tahmasebi, 《物理信息机器学习：地震波动方程》, *Geoscience Frontiers* 11, 1993 2001, 2020 , 第 pp 卷, 第–
页, 11月。[51] M. Rasht-Behesht、C. Huber、K. Shukla 和 G. E. Karniadakis, 《用于波传播和全波形反演的物理信息深度学习》, 2021年。[52] D. Mich´ea 和 D. Komatitsch, 《使用GPU显卡加速三维有限差分波传播代码：使用GPU加速波传播代码》, *Geophysical Journal International*, 第 pp 页, 第 no–no 页, 2010年5月。[53] S. A. Nørgaard、M. Sagebaum、N. R. Gauger 和 B. S. Lazarov, 《自动微分在拓扑优化中的应用》, *Structural and Multidisciplinary Optimization*, 第56卷, 第1135–1146页, 2017年11月。[54] C. B. Dilgen、S. B. Dilgen、D. R. Fuhrman、O. Sigmund 和 B. S. Lazarov, 《湍流的拓扑优化》, *Computer Methods in Applied Mechanics and Engineering*, 第331卷, 第363–393页, 2018年4月。[55] R. T. Q. Chen、Y. Rubanova、J. Bettencourt 和 D. Duvenaud, 《神经常微分方程》, 2019年12月。arXiv:1806.07366 [cs, stat].[56] R. J. Williams 和 D. Zipser, 《循环网络的基于梯度的学习算法及其计算复杂度》, 载于 *Backpropagation: theory、 architectures、 and applications*, 第433–486页, 美国：L. Erlbaum Associates Inc., 1995年1月。[57] I. Sutskever, *Training recurrent neural networks*。博士论文, 多伦多大学, 加拿大, 2013年。AAINS22066 ISBN-13:

9780499220660。[58] A. Chandrasekhar 和 K. Suresh，《TOuNN：使用神经网络进行拓扑优化》，*Structural and Multidisciplinary Optimization* 63，1135 1149，2021，第 pp 卷，第–页，2020年3月。[59] A. Wirgin，《逆问题犯罪》，2004年1月。arXiv:math-ph/0401050。[60] J. Parvizian、A. D¨uster 和 E. Rank，《有限单元法：固体力学中嵌入域问题的 h 与 p 扩展》，*Computational Mechanics*，第41卷，第121–133页，2007年9月。[61] G. J. Hicks，《使用凯泽窗sinc函数在有限差分格式中任意设置震源与检波器位置》，*GEOPHYSICS*，第67卷，第156–165页，2002年1月。[62] L. Hug、M. Potten、G. Stockinger、K. Thuro 和 S. Kollmannsberger，《基于实验测定II型断裂韧度的岩石混合型断裂三场相场模型》，*Engineering with Computers* 2022，7月。[63] T. Karras、T. Aila、S. Laine 和 J. Lehtinen，《用于提升质量、稳定性和多样性的GAN渐进式增长》，2018年2月。arXiv:1710.10196 [cs, stat].[64] A. D. Jagtap、K. Kawaguchi 和 G. E. Karniadakis，《自适应激活函数加速深度神经网络和物理信息神经网络中的收敛》，*Journal of Computational Physics*，第404卷，第109136页，2020年3月。[65] K. He、X. Zhang、S. Ren 和 J. Sun，《深入研究整流器：超越人类水平的ImageNet分类性能》，2015年2月。arXiv:1502.01852 [cs] 版本：1。[66] I. Goodfellow、Y. Bengio 和 A. Courville，*Deep Learning*。麻省理工学院出版社，2016年。http://www.deeplearningbook. org。[67] R. Pascanu、T. Mikolov 和 Y. Bengio，《循环神经网络训练的困难》，2013年2月。arXiv:1211.5063 [cs].[68] J. Zhang、T. He、S. Sra 和 A. Jadbabaie，《为什么梯度裁剪能加速训练：自适应性的理论解释》，2020年2月。arXiv:1905.11881 [cs, math].[69] S. K. Kumar，《深度神经网络中的权重初始化》，2017年5月。arXiv:1704.08863 [cs].[70] A. Fichtner，《全球尺度全波形层析成像的震源叠加数据降维》，载于*Full Seismic Waveform Modelling and Inversion*（A. Fichtner 编），《地球物理与环境力学和数学进展》，第281–299页，柏林、海德堡：Springer，2011年。[71] L. Herrmann、T. B¨urchner、F. Dietrich 和 S. Kollmannsberger，《关于神经网络用于全波形反演 [软件],的使用》*Mendeley Data*，2023年。[72] R. C. Gonzalez 和 R. E. Woods，*Digital image processing*。培生，第四版，全球版。