arXiv:2603.04827v1 [cs.LG] 5 Mar 2026

Kolmogorov-Arnold 网络的多层级训练

Ben S. Southworth* Jonas A. Actor† Graham Harper‡ Eric C. Cyr‡2026年3月6日

摘要

由于这类网络固有的函数复合缺乏结构保证，要通过算法加速常见神经网络架构的训练并不容易。与多层感知机（多层感知机）不同，Kolmogorov-Arnold 网络（KAN）通过在指定的基上展开学习到的激活函数，提供了更多结构。本文利用这一结构，提出实用算法并获得理论洞见，从而通过多层级训练加速 KAN 的训练。为此，我们首先通过线性基变换，确立采用样条基函数的 KAN 与采用幂 ReLU 激活函数的多通道多层感知机之间的等价性。随后，我们分析这种基变换如何影响相对于样条节点的梯度优化几何。KAN 的基变换促使我们采用多层次训练方法：通过对样条节点进行均匀细化，自然地定义一系列 KAN，并使用模型间解析的几何插值算子进行训练。该插值方案构成了一个“恰当嵌套层次结构”的架构，确保插值到细模型时能够保留在粗模型上取得的进展，而样条基函数的紧支撑则确保后续层级上的互补优化。数值实验表明，我们的多层级训练方法在准确率方面可比传统方法提升数个数量级，这些传统方法用于训练性能相当的 KAN 或多层感知机，尤其是在物理信息神经网络中。最后，本文展示了神经网络的原则性设计如何形成可加以利用的结构；在本例中，这种结构带来了能够显著提升训练性能的多层级算法。

1 引言

多层感知机（MLP） [42, 49, 45] 是一种经典的深度学习架构，利用仿射映射与非线性标量激活函数的组合。多层感知机架构作为模块，广泛应用于许多最先进的领域，包括但不限于变分自编码器 [34] 和 Transformer [62]。训练多层感知机，尤其是训练最先进架构的现代多通道和多头变体，是一项复杂的数值任务。这些方法通常依赖随机梯度下降的迭代变体，因而收敛速度相对较慢。相比之下，在经典数值方法中，用于求解数值偏微分方程（PDE）的多层级和多重网格方法，是求解线性与非线性方程最强大的方法之一，能够在 $\mathcal{O}(n)$次运算内求解稀疏 $n \times n$ 线性系统。

受这一成功的启发，已有不少尝试将多重网格思想拓展到机器学习领域，例如 [9, 20, 13, 32, 23, 16, 38, 18, 17]。深度神经网络与离散动力系统之间的联系 [21, 50, 10] 为将数值偏微分方程中的多层方法应用于神经网络训练提供了理论依据，因为残差网络的深度可以解释为时间离散化。基于这一解释， [9] 考虑了通过层内多层级进行嵌套迭代，并观察到有限的加速效果； [20, 13], 则将时间多重网格的概念应用于开发层并行训练策略，但所获得的加速源于并行化，而非算法本身。还有一些研究设计了明确融入多重网格结构的架构，例如 [32, 23, 16]。这些方法表明，多重网格原理有助于架构设计，但它们关注的是模型结构，而非训练算法，也未报告训练加速效果。 [38, 18], 研究了非线性多重网格与循环迭代背景下的严格多层级训练，并展示了有限的收敛性改进。最近

*洛斯阿拉莫斯国家实验室理论部，美国。
†桑迪亚国家实验室计算研究中心，美国。

Feischl 等人 [17] 提出了一个用于细化前馈神经网络的理论框架，并将其纳入优化过程。

然而，据我们所知，目前尚无多层级机器学习研究能像数值偏微分方程等其他领域那样实现算法加速。总体而言，这是由于缺乏层间具有良好逼近性质的多层级机器学习层次结构，也缺乏能够与所选粗化和插值相辅相成、针对网格的优化或“松弛”程序。在这一背景下，一个隐含的问题是，定义机器学习模型的粗表示相当困难。由于机器学习中的粗模型和细模型通常在同一维度空间中运行，因此无法从多重网格文献中明确推广逼近性质，以指导粗模型和传递算子的选择。概括而言，粗模型必须满足以下条件：(i) 比细模型更便宜地完成（近似）求解；(ii) 不与细模型的目标相冲突；此外，(iii) 能提供细模型难以捕捉或识别的校正或下降方向，即能与“松弛”或针对网格的优化方法互补。

作为多层感知机的替代架构，Kolmogorov-Arnold 网络（KAN） [40], 近年来在文献中日益受到关注，已被用于多种任务，包括计算机视觉 [8, 7, 11], 时间序列分析 [61, 15], 科学机器学习 [1, 59, 65, 35, 30, 44, 28, 48], 图分析 [33, 6], 等；有关 KAN 的不同版本、扩展和应用列表，参见 [54, 27] 及其中的参考文献；全面综述则见 [43]。文献中多次指出，多层感知机与 KAN 在结构上具有相似性，例如 [45, 29, 39, 24, 47]。由于二者相似，KAN 在学习任务中的表现往往可与多层感知机相媲美，具有相同的渐近复杂度和收敛速度 [19] ，并且在多种直接对比中表现相当 [67, 51, 66]。与多层感知机相比，KAN 尤其以以下特点著称：（i）可解释性更强，因为模型输出由解析函数复合构成；（ii）比传统多层感知机更能捕捉低正则性解和映射。然而，关于 KAN 的理论见解和实用算法，尚未能一致地展现这些特性并同时保持（或超越）多层感知机的性能，相关研究也仍显不足。本文旨在弥补这一差距，特别致力于推进训练策略，以利用多层级优化算法。

1.1 主要贡献

本文拓展了关于 KAN 的理论与实践见解，加深了我们对架构与优化器如何协同以提升训练效果和模型性能的认识。我们将这些见解转化为一项演示，展示多层方法如何显著改善经过恰当设计的神经网络的训练。为此，我们:

1. 利用 KAN 与多通道 MLP 之间的关系，在这两种架构之间引入基变换映射；2. 分析这种基变换如何改变梯度下降的动态过程；3. 引入多层次优化的恰当嵌套层次结构 概念，证明采用适当插值算子的 KAN 满足这一试探形式，并设计和展示一种受多重网格方法启发的相应多层次训练方法。

接下来将介绍本文的纲要，以及促使我们进行相关分析的技术步骤和原因。

1.2 论文其余部分

我们首先在第 2.1 节利用 B 样条与 ReLU 激活函数之间的关系，以多通道多层感知机的语言重新表述 KAN。我们将展示，在适当的线性基变换下，采用样条激活函数的 KAN 架构等价于某种形式的多通道多层感知机，其中偏置经过固定，以匹配样条节点。在第 2.2 节中，我们证明 KAN 与多通道多层感知机之间的线性基变换，恰好对应于在样条节点上对 r 阶导数算子进行的有限差分离散化。这一基变换立即带来了一种直接的非递归样条基KAN实现，其速度快于典型的 Cox-de Boor 递归形式（见第 2.3 节）。

尽管 KAN 与多通道多层感知机在基变换下作为前向算子是等价的，但二者基于梯度的训练会产生本质不同的权重演化。第 3 节从原空间与对偶空间的几何角度对这一点进行了抽象讨论：KAN 与多通道多层感知机之间的基变换矩阵的某个函数，会根据所选空间的不同，充当下降方法的预条件器。结合第 2.2 节的分析可以看出，微分算子的特征结构会显著改善训练动态：多通道多层感知机的表述会优先训练跨越样条节点的光滑函数。与多通道多层感知机相比，这使 KAN 更容易实现复杂函数这一富有表达力的模型。

在第4节中，我们利用第2节和第3节的数学公式，开发了一种高效的基于样条的 KAN 多层级训练框架。该框架建立在样条参数化所带来的结构之上，而多层感知机缺少这种结构。为此，我们提出了一个用于多层次优化的新概念：恰当嵌套层次结构，它可确保插值到细模型时不会抵消在粗模型上取得的进展。我们的传递算子以几何方式构建，自然形成恰当嵌套的层次结构，并加速了 [40] 中提出的网格传递方法，使其快到足以在训练期间构建或应用。随后，第3节所发展的理论表明，对恰当嵌套的粗模型和细模型应用基于梯度的优化具有互补性，这是任何成功的多重网格方法都必须满足的基本要求。总而言之，我们认为，恰当嵌套的层次结构与各层级互补的基于梯度的优化程序，是成功的多重网格方法所必需的组成部分。

在第 5 节中，我们通过数值实验表明，将多层级训练应用于函数回归和物理信息神经网络（PINN），能够显著提升准确率和效率。对于相同问题，我们还展示了，将多层级训练应用于等效的多通道 MLP 基，其缺乏层级互补的基于梯度的优化，所得改进实际上为零，与仅使用粗模型相比没有差别。这是因为细网格多通道多层感知机的训练会优先处理粗模型已经捕获的模态。

2 KAN 与多通道多层感知机

我们首先介绍以样条基表示的 KAN，然后说明线性变换如何将样条基中的每一层映射为等价的多通道 MLP。从历史和逼近理论的角度来看， [41, 56, 45, 57], KAN 和多层感知机都源于对柯尔莫哥洛夫叠加定理（KST）的拓展论证 [37], ；该定理也称为柯尔莫哥洛夫-阿诺德叠加定理。该定理指出，存在函数 $\phi_{pq} \in C([0,1])$，其中 $p = 1, \ldots, n$，以及$q = 1, \ldots, 2n+1$，使得对于任意函数 $f \in C([0,1]^n)$，都存在函数 $\varphi_q \in C(\mathbb{R})$，满足$f(x_1, \ldots, x_n) = \sum_{q=1}^{2n+1} \varphi_q \left(\sum_{p=1}^{n} \phi_{pq}(x_p) \right)$. 经典情况下， φ_q和 ϕ_{pq}在一个稠密点集上不可微，且仅为 H¨older 连续，或至多为利普希茨连续 [5, 56, 2]。受 KST 启发，KAN 将函数 φ_q 和 ϕ_{pq} 替换为以固定基表示的可学习函数，通常采用样条基 [40]。原始 KST 通过单层函数复合构造浅层网络，而 KAN 则使用更多层函数复合，以增加所得表示的深度。对于具有 P 个输入和Q 个输出的层 ℓ ，将 KAN 层的输出定义为 $x_q^{(\ell+1)} = \sum_{p=1}^{P} \phi_{pq}^{(\ell)}(x_p^{(\ell)})$，其中 $q = 1, \ldots, Q$，且 $\phi_{pq}^{(\ell)}$由数据训练得到。实际应用中，每个 $\phi_{pq}^{(\ell)}$ 都必须采用计算上可处理的形式表示。按照 [40]„ 我们将每个 $\phi_{pq}^{(\ell)}$ 展开为由 r阶 B 样条构成的基，从而得到有限维参数化，其中样条系数成为网络的可训练权重。选择该基是出于样条的逼近性质及其紧支撑，这些特性将在本文中得到充分利用。具体而言，给定一组节点，每个 $\phi_{pq}^{(\ell)}$都表示为 $\phi_{pq}^{(\ell)}(x) = \sum_{i=1-r}^{n-1} \widetilde{W}(_{qpi}^{\ell} b_i^{[r]})(x)$，其中 $\widetilde{W}_{qpi}^{(\ell)}$为可学习权重，而 $\{b_i^{[r-1]}\}_{i=1-r}^{n-1}$ 构成 B 样条基。

2.1 基变换

我们将样条基的结构形式化，以通过线性基变换建立其与使用 ReLU 激活函数的另一种表示之间的等价关系。令 $T = \{t_i\}_{i=1-r}^{n+r-1}$ 为一组严格有序的[1] 个 $n+2r-1$样条节点，其中 $t_i < t_{i+1}$，且 $t_0 = a$ 和 $t_n = b$。令 $\mathbb{P}^k$ 表示次数为 k的多项式空间。阶数为 r、节点为 T的样条空间，记作 $S_r(T)$，是由以下函数构成的集合

$$S_r(T) = \{f \in C^{r-2}([a,b]) \ : \ f|_{[t_i,t_{i+1}]} \in \mathbb{P}^{r-1},\, t_i \in T\}. \tag{1}$$

r阶样条空间由次数为 $r-1$的分段多项式组成（并受额外的光滑性约束）。对于 $r = 1$，定义 $b_i^{[1]}(x) = \begin{cases} 1 & x \in [t_i, t_{i+1}] \\ 0 & \text{否则} \end{cases}$；对于 $r > 1$，我们使用Cox-de Boor [14] 递推公式：

$$b_i^{[r]}(x) = \frac{x - t_i}{t_{i+r-1} - t_i} b_i^{[r-1]}(x) + \frac{t_{i+r} - x}{t_{i+r} - t_{i+1}} b_{i+1}^{[r-1]}(x). \tag{2}$$

每个函数 $b_i^{[r]}$ 支撑在区间 $[t_i, t_{i+r}]$上。我们另外定义一个ReLU^{r-1} 基，由函数张成

$$\psi_i^{[r]}(x) = \mathrm{ReLU}(x - t_i)^{r-1}. \tag{3}$$

现在我们可以重述一个经典的样条逼近结果，参见 [12]。

引理 1. $B_S^{[r]} = \{b_i^{[r]}\}_{i=1-r}^{n-1}$ 与 $B_R^{[r]} = \{\psi_i^{[r]}\}_{i=1-r}^{n-1}$ 均为 $S_r(T)$的基。

因此，它们之间存在一个线性基变换；将该基变换矩阵记为 $A^{[r]} \in \mathbb{R}^{(n+r-1)\times(n+r-1)}$，从而

$$B_S^{[r]} = A^{[r]} B_R^{[r]}, \tag{4}$$

其中 $B_S^{[r]}$ 和 $B_R^{[r]}$ 被理解为以基向量为列的方阵。矩阵 $A^{[r]}$ 具有结构且呈带状，在均匀节点间距的情形下有一个特别优雅的表达式：

引理 2. 对于 $r > 1$，令 $A^{[r-1]}$ 表示在阶为 r 的样条所用节点上构造的阶为 $r-1$ 的样条。则矩阵 $A^{[r]}$ 按元素定义为：

$$A_{ij}^{[r]} = \frac{1}{t_{i+r-1} - t_i} A_{ij}^{[r-1]} - \frac{1}{t_{i+r} - t_{i+1}} A_{i+1,j}^{[r-1]}, \tag{5}$$

其中我们定义 $A_{n+r}^{[r-1]}, := 0$，以处理最后一行 $i = n + r - 1$ 的特殊情形。

证明。 见B节。 □

推论 1（均匀节点）。 在均匀节点的情形下，间距为 $h = t_i - t_{i-1}$ 时， $A^{[r]}$ 为上三角托普利兹矩阵，其元素为

$$A_{ij}^{[r]} = \begin{cases} \frac{(-1)^{j-i}}{(j-i)!\,(r-j+i)!} \, \frac{r}{h^{r-1}} & i \le j \le i + r \\ 0 & else \end{cases}.$$

证明。 参见 B节。 □

现在我们说明，基变换 $A^{[r]}$ (4) 建立了 KAN 与某种形式的多通道 MLP 之间的等价关系。KAN架构的每一层在样条基 $B_S^{[r]}$ 下都具有存储为三阶张量 $\widetilde{W}^{(\ell)}$ 中参数/权重的样条系数，使得

$$\mathbf{x}_q^{(\ell+1)} = \sum_{p=1}^{P} \sum_{i=1-r}^{n-1} \widetilde{W}_{qpi}^{(\ell)} \, b_i^{[r]}(\mathbf{x}_p^{(\ell)}). \tag{6}$$

[1]我们假设样条节点严格有序，即不存在退化情形，这样可以简化所需的假设和定义。尽管作者认为，从理论上并无理由不能在这一框架下考虑退化样条，但要恰当地处理这些情况，需要进行大量严谨的技术工作，因此本文不作讨论。

定义 $W_{(\ell)} = \widetilde{W}^{(\ell)} \times_3 (A^{[r]})^T$，其中 $\times_3$ 表示三模张量积[2]；在索引层面， $W^{(\ell)}_{qpj} := \sum_{i=1-r}^{n-1} \widetilde{W}^{(\ell)}_{qpi} A^{[r]}_{ij}$。将 (4) 中的基变换和 $W_{(\ell)}$ 的定义代入 (6) 可得

$$\mathbf{x}_q^{(\ell+1)} = \sum_{p=1}^{P} \sum_{i=1-r}^{n-1} \widetilde{W}^{(\ell)}_{qpi} \left(\sum_{j=1-r}^{n-1} A^{[r]}_{ij} \operatorname{ReLU}(\mathbf{x}_p^{(\ell)} - t_j)^{r-1} \right) \tag{7a}$$

$$= \sum_{p=1}^{P} \sum_{j=1-r}^{n-1} W^{(\ell)}_{qpj} \operatorname{ReLU}(\mathbf{x}_p^{(\ell)} - t_j)^{r-1}. \tag{7b}$$

根据 A节中概述的记号变换，我们注意到，式（7）是一个具有 ReLU^{r-1} 激活的等价稠密多通道多层感知机层。综上，我们得到以下结果。

引理3（KAN与多通道多层感知机的等价性）。 对于形如（6）的单层 *KAN*，其权重三阶张量为 $\widetilde{W}^{(\ell)}$，它等价于形如（7）的单层多通道多层感知机，其权重三阶张量为 $W_{(\ell)} = \widetilde{W}^{(\ell)} \times_3 (A^{[r]})^T$。

现在考虑一个由多层组成的完整网络。为简化记号，并结合第3节的内容，令我们将所有层的权重进行向量化，并将**u** 记为样条基中的完整权重向量，将**w** 记为 ReLU 基中的权重向量；二者分别为**u**、$\mathbf{w} \in \mathbb{R}^{N_w}$。根据上述讨论，通过对 $A^{[r]}$ 进行适当的拼接和重新排序/展平，我们可以定义基变换矩阵 A，使得

$$A^T \mathbf{u} = \mathbf{w}.$$

注意，此处的基变换将样条基中的权重映射为 ReLU 基中的权重，这与函数基变换的方向相反 (4)。假设权重**u**和**w** 的向量化操作按层、输出神经元、输入神经元，最后按样条节点的顺序排列。按照此顺序，所得的 A 在前三个维度（层、输出神经元和输入神经元）上为块对角矩阵，其对角块是在样条节点上的适当维度的（$A^{[r]}$）。

2.2 $A^{[r]}$的特征结构

现在，我们证明以下引理，说明 $A^{[r]}$ 作为某些微分算子的离散近似时所具有的结构。

引理 4。 除以 $\pm(r-1)!/h$ 进行常数缩放外（符号取决于 r 是奇数还是偶数）， $A^{[r]}$ 是网格间距为 h的一维均匀网格上 r 阶导数的前向有限差分近似，并严格施加零狄利克雷边界条件。除边界节点和常数缩放外，$(A^{[r]})^T A^{[r]}$也是托普利茨矩阵，并且是网格间距为 h的一维均匀网格上$(2r)$阶导数的有限差分近似。

证明。 考虑样条次数为 r 的模板 $\mathbf{v}^{(r)}$，从对角线开始并延伸至上三角索引。令对角线索引对应于零，并令 $\mathbf{v}_i^{(r)}$ 表示第 i 个模板索引。那么

$$\mathbf{v}_i^{(r)} := \frac{(-1)^i}{i!\,(r-i)!} \frac{r}{h^{r-1}} = \frac{1}{(r-1)! h^{r-1}} \left[(-1)^i \binom{r}{i} \right], \quad i \le r. \tag{8}$$

No我们回顾一下以 x 为中心的 $f^{(r)}(x)$ 的一般前向有限差分，其表达式为 [5 2]

$$f^{(r)}(x) \approx \frac{1}{h^r} \sum_{i=0}^{r} (-1)^{r-i} \binom{r}{i} f(x + ih). \tag{9}$$

至此完成了 $A^{[r]}$的证明，其中零狄利克雷边界条件对应于最终矩阵各行中的截断托普利茨结构。关于$(A^{[r]})^T A^{[r]}$的托普利茨性质及相应的模板值，参见

[2]有关此运算的更多详情，请参见 [36]。

[55, 引理 23]。与上述讨论一致，内部托普利兹算子的模板在至多相差一个常数全局缩放的意义下，取一般中心有限差分对第(2r)阶导数的逼近形式，

$$f^{(2r)}(x) \approx \frac{1}{h^{2r}} \sum_{i=0}^{2r} (-1)^i \binom{2r}{i} f(x + (r-i)h). \tag{10}$$

□

引理 4 证明了 KAN 架构中 ReLU 基与样条基之间的离散微分关系。例如，忽略边界节点，对于 $r = 1$，我们有 $(A^{[r]})^T A^{[r]}$ 对应于一维各向同性扩散的$[1, -2, 1]$ 模板。这立即让我们对 $(A^{[r]})^T A^{[r]}$ 作为算子时的作用有了深刻直觉。下面考虑 (2r)阶导数算子 D^{2r} 的特征值：$= \frac{d^{2r}}{dx}$，并考虑定义域为 $x \in [0, 1]$、具有周期边界条件 $f^{(k)}(0) = f^{(k)}(1)$ 的简化情形，其中$k \in \{0, ..., 2r-1\}$。考虑作为特征向量的傅里叶基

$$f_i(x) := e^{-2\pi i i x} \quad i \in \mathbb{Z}. \tag{11}$$

求 $2r$ 次导数可得

$$f_i^{(2r)}(x) = D^{2r} e^{-2\pi i i x} = (-2\pi i i)^{2r} e^{-2\pi i i x} = (-1)^r (2\pi i)^{2r} e^{-2\pi i i x}, \tag{12}$$

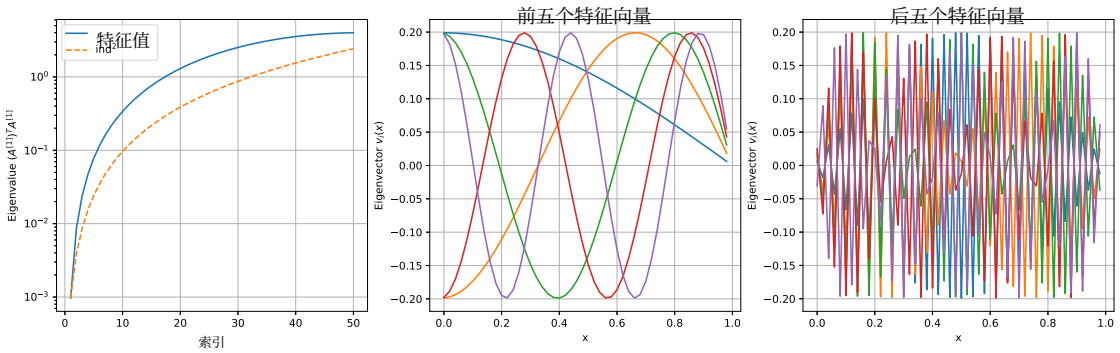
对于 $i \in \mathbb{Z}$。因此， D^{2r} 的特征值为 $\lambda \in \{(-1)^r(2\pi i)^{2r}\}$，其中 $i \in \mathbb{Z}$；特征值的模 λ 与对应特征向量 $f_i(x) = e^{-2\pi i i x}$ 的傅里叶频率直接相关。

转向离散情形和样条换基后，$(A^{[r]})^T A^{[r]}$ 不会施加周期边界条件；而且由于正规方程形式的性质，特征值将严格为正，而不像连续分析中那样随 r 交替变号。这是因为，与式 (10) 中的有限差分模板相比，$(A^{[r]})^T A^{[r]}$ 近似的是 (2r)阶导数的交替符号。不过，我们预期其总体性质仍然成立。首先，特征值的取值范围很大，并且（除去常数缩放外）对于 n 个样条节点，其形式大致为 ℓ^{2r}，其中特征值指标为 $\ell \in [1, n]$。其次，特征模态会按照光滑度依次排列：最小特征值对应最光滑的特征模态，最大特征值则对应样条网格上振荡最强的特征模态。不过，由于边界条件并非周期性，这组特征基并非严格的傅里叶基。图 1 展示了 $r = 1$ 和 $r = 3$ 的上述性质。接着，图 2a 通过考察最小与最大特征值之比，展示了 $(A^{[r]})^T A^{[r]}$ 对样条网格上的光滑模态和振荡模态进行缩放时的差异。鉴于特征值近似遵循 ℓ^{2r}，我们预计该比值按 n^{2r} 缩放，这一点也在图 2a 中得到验证。因此，即使样条节点数量 $n = 10$ 很少，振荡最强的模态也比最光滑的模态缩放得更强：对于 $r = 1$，强 $100\times$ 倍；对于 $r = 3$，强 $10^6\times$ 倍。为了更直观地展示几何结构，我们还在图 2b 中绘出了离散特征向量的符号变化次数。在这里，我们看到了预期的简单结构，且与 r 无关：$(A^{[r]})^T A^{[r]}$ 中与最小特征值对应的特征向量没有符号变化，而与最大特征值对应的特征向量有 $n-1$ 次符号变化。自然地，$(A^{[r]}(A^{[r]})^T)^{-1}$ 遵循完全相反的顺序。

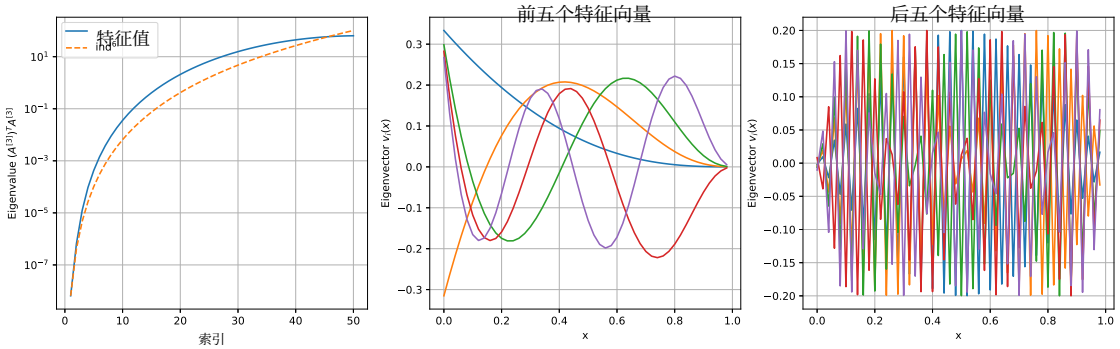
我们将在第 3 节中展示，这些传递算子的特征结构对训练动态有着重要影响。

2.3 计算成本

虽然 $B_R^{[r]}$ 和 $B_S^{[r]}$ 这两种基在输出方面等效，但在计算每一层的输出时，其计算成本有所不同。研究发现，Cox-de Boor公式的计算开销较大 [46], ，而采用 $B_R^{[r]}$ 进行重新表述后，可显著加速底层计算图。对于单层而言，标准 B样条基的一次前向传播需要 $O(PQ(nr + r^2))$ 次浮点运算；基于ReLU的形式只需阶数为$O(PQ(n + r))$ 的运算，因此浮点运算次数（FLOPs）可加速样条次数倍。这样便无需像 [46, 53]那样实现不同版本的样条激活函数。基变换运算，即乘以 A，每层只需 $O(nr)$ 次运算，因为 A 是一个带状矩阵，其

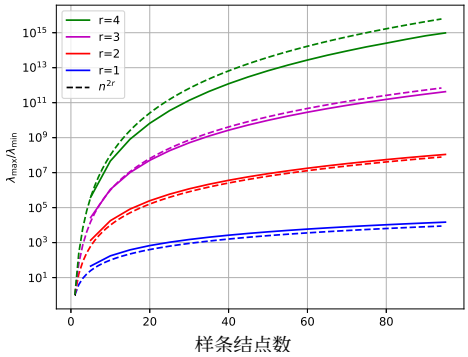


(a) $r = 1$

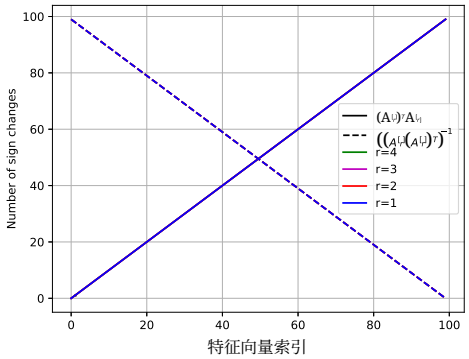


(b) $r = 3$

图 1：$(A^{[r]})^T A^{[r]}$ 的特征值按从小到大排列（左），以及对应的前五个（中）和后五个（右）特征向量，展示了 $n = 50$ 个样条节点和示例阶数 $r \in \{1, 3\}$。



(a) $(A^{[r]})^T A^{[r]}$ 的最大特征值与最小特征值之比随样条节点数变化，展示了 $r \in [1, 4]$。该比值决定了基于梯度下降的优化中对应模态的加权。



(b) 离散特征向量中的符号变化次数随特征向量索引的变化，对于 $n = 100$ 个节点、$r \in [1, 4]$，且特征值按升序排列。展示了 $(A^{[r]})^T A^{[r]}$ 和 $(A^{[r]}(A^{[r]})^T)^{-1}$。

图 2：不同样条阶数下特征值和特征向量性质的比较。

$r + 1$ 个非零对角线（显式构造见附录 B）。因此，对于较小的样条阶数 r，即使 P 和 Q 的值适中，其计算成本相较于与可学习权重的收缩也可忽略不计。对于一组预先确定的样条节点，矩阵 A 还可以通过卷积模板实现，这对于均匀节点尤其高效。

为说明这些结果，我们使用 (6) 和 (7) 的直接实现训练一个简单架构，该架构包含三个宽度为 64 的隐藏层，其中样条基函数通过标准的 Cox-de Boor 递归形式实现。我们针对一系列样条阶数 r 和节点数量 n 测量每个训练轮次的挂钟时间，并将每个实验重复 10 次，以平滑挂钟测量中的噪声；我们的挂钟测量仅包括通过网络的前向和反向传播，不包括数据加载以及验证数据集评估所用的时间。现代图形处理器上的线性代数运算利用了硬件架构，这使得挂钟测量变得复杂，但在图 3 中，我们

仍然看到挂钟时间有显著改善，且该改善随 r 增大。

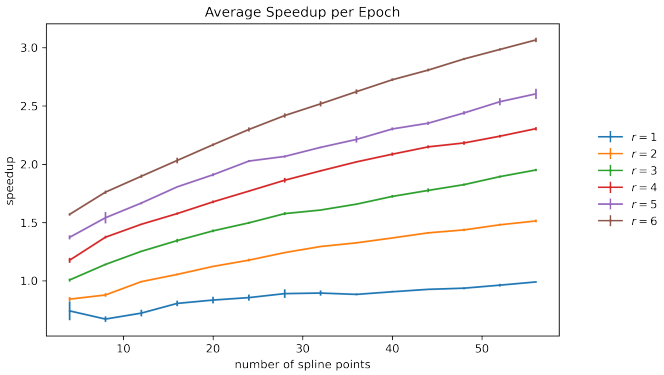


图 3：通过应用 ReLU^r 激活函数后再乘以基变换矩阵，与计算样条函数的 Cox-de Boor递推公式相比，评估一层所获得的加速效果。误差棒表示在 10 个实例上计算的 1 个标准差。

3 梯度下降与基的选择

尽管在基变换下，KAN 和多通道多层感知机作为前向算子是等价的，但它们在训练过程中基于梯度的训练会呈现根本不同的动态；我们通过将基变换运算与预条件中的标准结果联系起来，展示了这一点。

作为对第2节中各类架构的补充，我们将训练问题的符号表示形式化。我们考虑训练一个神经网络，其输入数据维度为 $d_{\text{in}} \in \mathbb{N}$，输出数据维度为$d_{\text{out}} \in \mathbb{N}$；训练数据是由 k 个输入—输出数据对组成的展平批次 $\{\mathbf{x}, \mathbf{y}\}$，其中 $\mathbf{x} \in \mathbb{R}^{kd_{\text{in}}}$，$\mathbf{y} \in \mathbb{R}^{kd_{\text{out}}}$。令 $f(\mathbf{x}; \mathbf{w}) : \mathbb{R}^{kd_{\text{in}}} \times \mathbb{R}^{N_w} \mapsto \mathbb{R}^{kd_{\text{out}}}$表示权重为 $\mathbf{w}$时基于ReLU的网络对批次输入数据 $\mathbf{x}$的作用，并令 $\mathcal{L} : \mathbb{R}^{kd_{\text{in}}} \times \mathbb{R}^{kd_{\text{out}}} \times \mathbb{R}^{N_w} \mapsto \mathbb{R}$ 表示给定权重下数据上的标量值损失。于是，对权重进行相应的无约束优化可表示为 $\min_{\mathbf{w}\in\mathbb{R}^n} \mathcal{L}(f(\mathbf{x}; \mathbf{w}), \mathbf{y})$。

我们将基变换引入这个优化问题，并定义一个非线性函数$g(\mathbf{x}; \mathbf{u}) : \mathbb{R}^{kd_{\text{in}}} \times \mathbb{R}^{N_w} \mapsto \mathbb{R}^{kd_{\text{out}}}$，使得 $g(\mathbf{x}; \mathbf{u}) = f(\mathbf{x}; A^T\mathbf{u}) = f(\mathbf{x}; \mathbf{w})$，对应于基于样条的KAN。然后，我们可以等价地最小化

$$\min_{\mathbf{w}\in\mathbb{R}^n} \mathcal{L}(f(\mathbf{x}; \mathbf{w}), \mathbf{y}) = \min_{\mathbf{u}\in\mathbb{R}^n} \mathcal{L}(g(\mathbf{x}; \mathbf{u}), \mathbf{y}). \tag{13}$$

令 $\nabla_{\mathbf{w}}\mathcal{L} \in \mathbb{R}^{N_w}$ 和 $\nabla_{\mathbf{u}}\mathcal{L} \in \mathbb{R}^{N_w}$ 分别表示关于变量对 $\{\mathbf{w}\ f\}$ 或 $\{\mathbf{u}\ g\}$ 的标准 ℓ^2-梯度（可通过反向传播轻松计算）。对于梯度下降，我们关注的是表示原空间与对偶空间的基。如果我们在原空间与对偶空间中分别仅使用 $\mathbf{w}$ 或 $\mathbf{u}$ 基，就会得到标准的、可能经过预条件的梯度下降迭代：

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \eta D_{\mathbf{w}} \nabla_{\mathbf{w}}\mathcal{L}, \tag{14a}$$
$$\mathbf{u}_{k+1} = \mathbf{u}_k - \eta D_{\mathbf{u}} \nabla_{\mathbf{u}}\mathcal{L}, \tag{14b}$$

对于线性预条件子 $D_{\mathbf{w}}, D_{\mathbf{u}}$，例如来自 ADAM 或 LBFGS（或者 $D = I$，用于梯度下降）。现在考虑基变换 $\mathbf{w} \mapsto A^T\mathbf{u}$或 $\mathbf{u} \mapsto A^{-T}\mathbf{w}$。令 $J_f := \partial f/\partial \mathbf{w} \in \mathbb{R}^{k\times N_w}$和 $J_g := \partial g/\partial \mathbf{u} \in \mathbb{R}^{k\times N_w}$表示 f和 g关于其自然变量的雅可比矩阵。由于 $g(\mathbf{x}; \mathbf{u}) = f(\mathbf{x}; A^T\mathbf{u})$，我们有 $J_g = \frac{\partial f}{\partial \mathbf{w}\partial \mathbf{u}} = J_f A^T$. 令 $\nabla_m\mathcal{L} \in \mathbb{R}^k$表示 $\mathcal{L}$关于 k个数据样本的模型输出的导数。由式 (13)，我们可以在$\mathbf{w}$空间中计算梯度：

$$\nabla_{\mathbf{w}}\mathcal{L} := \left(\frac{\partial f}{\partial \mathbf{w}}\right)^T \frac{\partial \mathcal{L}}{\partial f} = J_f^T \nabla_m \mathcal{L}. \tag{15}$$

同样地，我们可以在$\mathbf{u}$空间中计算梯度，并得到对偶（梯度）空间中的基变换：

$$\nabla_{\mathbf{u}}\mathcal{L} := \left(\frac{\partial g}{\partial \mathbf{u}}\right)^T \frac{\partial \mathcal{L}}{\partial g} = J_g^T \nabla_m \mathcal{L} = A J_f^T \nabla_m \mathcal{L} = A \nabla_{\mathbf{w}}\mathcal{L}. \tag{16}$$

将$\mathbf{w} \mapsto A^T\mathbf{u}$和 $\nabla_{\mathbf{w}}\mathcal{L} \mapsto A^{-1}\nabla_{\mathbf{u}}\mathcal{L}$代入（14a），并将$\mathbf{u} \mapsto A^{-T}\mathbf{w}$和 $\nabla_{\mathbf{u}}\mathcal{L} \mapsto A\nabla_{\mathbf{w}}\mathcal{L}$代入（14b），我们分别得到预条件梯度下降迭代式，

$$\mathbf{u}_{k+1} = \mathbf{u}_k - \eta A^{-T} D_{\mathbf{w}} A^{-1} \nabla_{\mathbf{u}}\mathcal{L}, \tag{17a}$$

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \eta A^{T} D_{\mathbf{u}} A \nabla_{\mathbf{w}}\mathcal{L}, \tag{17b}$$

其中，我们在基变换过程中保留了原有的预条件器。这一基变换也可以看作是将$\mathbf{w}$空间的几何结构施加到$\mathbf{u}$空间上。通过 A^T、$\mathbf{u} \mapsto A^{-T}\mathbf{u}$进行基变换，会在 U空间中诱导出拉回度量，$\langle \mathbf{x},\ \mathbf{y}\rangle_U := \mathbf{y}^T AA^T \mathbf{x}$；其中令 $D_{\mathbf{u}} = I$，现在梯度是相对于 (AA^T)-内积而言的。回顾以下结果 [4]:

引理 5。 考虑 $\mathcal{L}$： $\mathbb{R}^n \mapsto \mathbb{R}$，并令 $\nabla\mathcal{L}(\mathbf{x})$ 表示 L 在点 $\mathbf{x} \in \mathbb{R}^n$处相对于 ℓ^2-内积的梯度。令 $M \in \mathbb{R}^{n\times n}$为一个对称正定矩阵。则 $\mathcal{L}$ 相对于由 M诱导的内积的梯度为

$$\nabla_M \mathcal{L}(\mathbf{x}) = M^{-1}\nabla\mathcal{L}(\mathbf{x}). \tag{18}$$

我们看到，对于 $D_{\mathbf{u}} = I$，(17a) 是相对于 $\{\mathbf{u}\ g\}$、在 (AA^T)-内积下的梯度下降，它源于将 W空间的几何结构施加到 U空间上。类似地，对于 $D_{\mathbf{w}} = I$，(17b) 等价于在 $\{\mathbf{w}\ f\}$中、相对于 $(A^TA)^{-1}$内积的梯度下降，它源于拉回度量将 U空间的几何结构施加到 W空间上。

为完整起见，也可以混合使用不同的空间。假设我们在$\mathbf{w}$ 中迭代，但考虑相对于$\mathbf{u}$的梯度。此类迭代的形式为$\mathbf{w}_{k+1} = \mathbf{w}_k - \eta D_{\mathbf{u}}\nabla_{\mathbf{u}}\mathcal{L}$。代入 $\nabla_{\mathbf{u}}\mathcal{L} \mapsto A\nabla_{\mathbf{w}}\mathcal{L}$或$\mathbf{w} \mapsto A^T\mathbf{u}$后，我们得到迭代$\mathbf{w}_{k+1} = \mathbf{w}_k - \eta D_{\mathbf{u}} A\nabla_{\mathbf{w}}\mathcal{L}$或$\mathbf{u}_{k+1} = \mathbf{u}_k - \eta A^{-T} D_{\mathbf{u}}\nabla_{\mathbf{u}}\mathcal{L}$。类似地，我们可以在$\mathbf{u}$ 中迭代，但考虑相对于$\mathbf{w}$的梯度。此类迭代的形式为$\mathbf{u}_{k+1} = \mathbf{u}_k - \eta D_{\mathbf{w}}\nabla_{\mathbf{w}}\mathcal{L}$。代入 $\nabla_{\mathbf{w}}\mathcal{L} \mapsto A^{-1}\nabla_{\mathbf{u}}\mathcal{L}$ 或$\mathbf{u} \mapsto A^{-T}\mathbf{w}$ 后，我们得到迭代$\mathbf{u}_{k+1} = \mathbf{u}_k - \eta D_{\mathbf{w}} A^{-1}\nabla_{\mathbf{u}}\mathcal{L}$ 或$\mathbf{w}_{k+1} = \mathbf{w}_k - \eta A^T D_{\mathbf{w}}\nabla_{\mathbf{w}}\mathcal{L}$。请注意，在迭代所用的同一基中计算梯度，会得到采用对称正定预条件器的预条件梯度下降，而预条件器与诱导内积有关。相比之下，若相对于不同于迭代所用基的另一基来计算梯度，则会得到可能非对称的预条件下降法。这些方法对应于对偶配对的修改，它规定了每个对偶向量（梯度）与原向量之间的唯一映射。对偶配对由一个可逆但不一定对称正定的矩阵表示；在此情况下，具体来说，该矩阵由每个方程中起首的预条件算子给出（假设我们也不考虑修改内积）。

总的来说，我们有四种不同的迭代方式，它们在$\mathbf{u}$ 或$\mathbf{w}$ 中的等价实现见表 1。请注意，在本文余下部分中，出于可解释性的考虑，我们将在样条空间$\mathbf{u}$ 中进行迭代，并考察由$\mathbf{u}$ 几何和梯度 (14b) 或$\mathbf{w}$ 几何和梯度 (17a) 所产生的迭代；前者是 KAN 中自然产生的梯度下降。

几何	梯度	空间	精确更新
$\mathbf{w}$	$\nabla_{\mathbf{w}}\mathcal{L}$	$\mathbf{w}$	$\mathbf{w}_{k+1} = \mathbf{w}_k - \eta D_{\mathbf{w}}\nabla_{\mathbf{w}}\mathcal{L}$
		$\mathbf{u}$	$\mathbf{u}_{k+1} = \mathbf{u}_k - \eta A^{-T} D_{\mathbf{w}} A^{-1}\nabla_{\mathbf{u}}\mathcal{L}$
$\mathbf{w}$	$\nabla_{\mathbf{u}}\mathcal{L}$	$\mathbf{w}$	$\mathbf{w}_{k+1} = \mathbf{w}_k - \eta D_{\mathbf{u}} A\nabla_{\mathbf{w}}\mathcal{L}$
		$\mathbf{u}$	$\mathbf{u}_{k+1} = \mathbf{u}_k - \eta A^{-T} D_{\mathbf{u}}\nabla_{\mathbf{u}}\mathcal{L}$
$\mathbf{u}$	$\nabla_{\mathbf{u}}\mathcal{L}$	$\mathbf{w}$	$\mathbf{w}_{k+1} = \mathbf{w}_k - \eta A^T D_{\mathbf{u}} A\nabla_{\mathbf{w}}\mathcal{L}$
		$\mathbf{u}$	$\mathbf{u}_{k+1} = \mathbf{u}_k - \eta D_{\mathbf{u}}\nabla_{\mathbf{u}}\mathcal{L}$
$\mathbf{u}$	$\nabla_{\mathbf{w}}\mathcal{L}$	$\mathbf{w}$	$\mathbf{w}_{k+1} = \mathbf{w}_k - \eta A^T D_{\mathbf{w}}\nabla_{\mathbf{w}}\mathcal{L}$
		$\mathbf{u}$	$\mathbf{u}_{k+1} = \mathbf{u}_k - \eta D_{\mathbf{w}} A^{-1}\nabla_{\mathbf{u}}\mathcal{L}$

表 1：不同参数化和梯度表示下的预条件梯度下降更新规则。我们将初始预条件器设定为与梯度保持一致，例如 $\nabla_{\mathbf{u}}$ 的预条件器为 $D_{\mathbf{u}}$。基变换所诱导的预条件，结合第 2.2 节的谱理论，为理解并确保下一节提出的多层层次结构中每一层的互补优化或“松弛”提供了一个框架（见第4.2节）。

注释 1. 根据 A 的结构以及第*3*节推导出的每个区块 $A^{[r]}$，我们可以界定神经切线核（*NTK*）的谱半径，从而表明，尽管基变换会引入预条件，仍可对上述推导出的 *ReLU* 和样条基采用基于梯度的优化方法，并使用相近的学习率。不过，由此产生的训练动态将有显著差异；有关这些矩阵的 *NTK* 的更多信息，请参阅补充材料。

4 KAN 的多层级训练

我们基于 KAN 的基变换和预条件结果，利用样条参数化带来的额外结构，构建了一个高效的基于样条的 KAN 多层级训练框架。借助上述预条件结果，我们可以在自然样条基下理解多通道多层感知机；这种基提供了一个函数框架，便于构建具有清晰尺度层次结构和传递算子的模型。采用样条基函数和几何传递算子的 KAN 可提供适当嵌套的层次结构，实现不同程度的细化和互补的层级特定优化。

4.1 一般多层形式

设 $\mathcal{P}:\mathbb{R}^{N_c}\mapsto\mathbb{R}^{N_f}$ 是从维度为 N_c 的粗权重空间到维度为 N_f 的细权重空间的线性插值和基变换。在粗空间 $\mathbf{u}^{(c)}$ 上进行优化，我们得到梯度更新

$$\mathbf{u}_{k+1}^{(c)}=\mathbf{u}_k^{(c)}+\eta\mathcal{P}^T\nabla\mathcal{L}(\mathcal{P}\mathbf{u}_k^{(c)}). \tag{19}$$

假设 $\nabla\mathcal{L}(\mathbf{u})=L\mathbf{u}$ 对于线性算子 L 成立。那么这可化为

$$\mathbf{u}_{k+1}^{(c)}=\mathbf{u}_k^{(c)}+\eta\mathcal{P}^TL\mathcal{P}\mathbf{u}_k^{(c)}, \tag{20}$$

这正是对 Galerkin 粗网格算子 $L_c:=\mathcal{P}^TL\mathcal{P}$ 进行的理查森迭代。如果我们进一步在 $\mathcal{P}\mathbf{u}^{(c)}$ 中加入对角缩放，使得 L_c 的对角线为单位值，就得到应用于 Galerkin 粗网格算子的雅可比松弛迭代，这正是代数多重网格粗网格校正中的计算内核。[3]

回到优化问题，考虑模型 $g(\mathbf{x};\mathbf{u})$：$\mathbb{R}^{kd_{\text{in}}}\times\mathbb{R}^{N_w}\mapsto\mathbb{R}^{kd_{\text{out}}}$ 以及损失 $\mathcal{L}$：$\mathbb{R}^{kd_{\text{in}}}\times\mathbb{R}^{kd_{\text{out}}}\times\mathbb{R}^{N_w}\mapsto\mathbb{R}$。为了确保粗层级更新能够良好近似细层级目标函数，一种自然的方法是通过以下方式定义粗子空间优化问题

$$\min_{\mathbf{u}^{(c)}\in\mathbb{R}^{N_c}}\mathcal{L}(g(\mathbf{x};\mathcal{P}\mathbf{u}^{(c)}),\mathbf{y}). \tag{21}$$

这种解释为在权重空间中进行粗化提供了一个框架，同时确保良好的粗近似（这是多重网格的关键组成部分，而目前几乎所有多层机器学习方法都缺少这一点）。在实践中，我们不希望每一层的计算成本都与最细层一样高；由于内部模型 g 的规模，梯度会通过它反向传播，因此在（21）中直接照搬的做法会导致这种情况。不过，这确实为设计多层层次结构和考察逼近性质提供了一个自然目标。考虑一个两层层次结构。根据（21）中的定义，我们为在函数近似意义上恰当嵌套的层次结构，定义粗层架构与细层架构之间的自然关系，以及相应的传递算子。

定义 1（恰当嵌套层次结构）。令 $g_f(\mathbf{x};\mathbf{u}^{(f)}):\mathbb{R}^{kd_{\text{in}}}\times\mathbb{R}^{N_f}\mapsto\mathbb{R}^{d_{\text{out}}}$ 表示细层算子对展平输入向量 $\mathbf{x}\in\mathbb{R}^{kd_{\text{in}}}$ 在细网格权重 $\mathbf{u}^{(f)}\in\mathbb{R}^{N_f}$ 下的作用，并令 $g_c(\mathbf{x};\mathbf{u}^{(c)}):\mathbb{R}^{kd_{\text{in}}}\times\mathbb{R}^{N_c}\mapsto\mathbb{R}^{d_{\text{out}}}$ 表示粗层算子对展平输入向量 $\mathbf{x}\in\mathbb{R}^{kd_{\text{in}}}$ 在粗网格权重 $\mathbf{u}^{(c)}\in\mathbb{R}^{N_c}$ 下的作用。令 $\mathcal{P}:\mathbb{R}^{N_c}\mapsto\mathbb{R}^{N_f}$ 为从粗权重空间到细权重空间的线性插值算子。若满足以下条件，则称 $\{g_f,g_c,\mathcal{P}\}$ 为一个恰当嵌套层次结构

$$g_c(\mathbf{x};\mathbf{u}^{(c)})=g_f(\mathbf{x};\mathcal{P}\mathbf{u}^{(c)}), \tag{22}$$

也就是说，适当嵌套的层次结构要求细层算子在权重插值下精确保留粗层算子的作用。

该定义保证，权重插值不会抵消通过粗层优化所取得的进展，因为细模型及其对应的损失将与粗模型完全一致。如果我们满足上述定义，就可以构建基变换层次结构，而无需在每一层都以细层代价评估网络。特别地，我们满足以下命题。

[3]通过在非标准内积或对偶配对下考虑梯度，可以实现带有限制算子 $\mathcal{R}\neq\mathcal{P}^T$ 的 Petrov-Galerkin 传递算子。本文不考虑这类情况。

命题*1*（子空间优化）。设 g_f、g_c：$\mathbb{R}^{d_{\text{in}}} \mapsto \mathbb{R}^{d_{\text{out}}}$ 与 $\mathcal{P}$：$\mathbb{R}^{N_c} \mapsto \mathbb{R}^{N_f}$ 如定义 *1* 所述，并定义关于模型输出相同的细损失函数与粗损失函数 $\mathcal{L}$：$\mathbb{R}^{kd_{\text{in}}} \times \mathbb{R}^{kd_{\text{out}}} \times \mathbb{R}^{N_f} \mapsto \mathbb{R}$、$\mathcal{L}_c$：$\mathbb{R}^{kd_{\text{in}}} \times \mathbb{R}^{kd_{\text{out}}} \times \mathbb{R}^{N_c} \mapsto \mathbb{R}$，也就是说，若 $g_c = g_f$，$\mathcal{L}_c(\cdot;\cdot,\cdot) = \mathcal{L}(\cdot;\cdot,\cdot)$，则 $\mathcal{L}_c(g_c(\mathbf{x};\mathbf{u}^{(c)}),\mathbf{y}) = \mathcal{L}(g_f(\mathbf{x};\mathcal{P}\mathbf{u}^{(c)}),\mathbf{y})$。则

$$\min_{\mathbf{u}^{(c)}} \mathcal{L}_c(g_c(\mathbf{x};\mathbf{u}^{(c)}),\mathbf{y}) = \min_{\mathbf{u}^{(c)}} \mathcal{L}(g_f(\mathbf{x};\mathcal{P}\mathbf{u}^{(c)}),\mathbf{y}). \tag{23}$$

这是一个简单的结果，它形式化地说明了恰当嵌套层次结构的目的——在粗空间中优化粗粒度损失 $\mathcal{L}_c$ 等价于将权重限制在一个子空间内，对细粒度损失进行优化。然而，我们必须谨慎进行粗化，以确保 (22) 可行。如果采用文献中多次讨论过的层内粗化，例如 [20, 13, 38], ，则可能无法满足此类约束。对于任意数量的层级，我们在算法1中定义了嵌套多层优化。请注意，本文不考虑恰当的多层循环迭代。非线性多重网格和多重网格优化一向对参数十分敏感，这将留待未来研究。

算法1 嵌套多层优化

输入： 层次结构 $\{g_k\}_{k=1}^{K}$，传递算子 $\{\mathcal{P}_k^{k-1}\}_{k=2}^{K}$，训练数据 $\mathbf{x}$

输出： 最细层问题的解 $\mathbf{u}_1$，$\min_{\mathbf{u}_1} \mathcal{L}(g_1(\mathbf{x};\mathbf{u}_1))$。

初始化 $\mathbf{u}_K$ {在最粗层初始化权重}

对于 $k = K$ 到 2: **执行**

  求解 $\min_{\mathbf{u}_k} \mathcal{L}(g_k(\mathbf{x};\mathbf{u}_k))$ {使用初始猜测 $\mathbf{u}_k$ 优化第 k 层}

  $\mathbf{u}_{k-1} \leftarrow \mathcal{P}_k^{k-1}\mathbf{u}_\ell$ {将解插值到更细层级，作为初始猜测}

结束循环

求解 $\min_{\mathbf{u}_1} \mathcal{L}(g_1(\mathbf{x};\mathbf{u}_1))$ {在最细层上进行优化，初始猜测为 $\mathbf{u}_1$}

返回 $\mathbf{u}_1$

4.2 KAN 中的传递算子

利用样条的多分辨率特征构建不同分辨率的 KAN，这一方法由 [40] 提出。然而， [40] 中的细化并非几何细化；相反，他们在任意大小的网格之间进行插值。为此，他们针对每个隐藏节点求解一个大小为 $N_{\text{data}} \times (n+r-1)$ 的最小二乘问题，这一计算的成本高得令人难以承受，这一点并不意外。作者认为，这种方法速度太慢，不适用于实际应用。相比之下，先进行几何细化，再利用多重网格文献 [25, 22] 中的限制与延拓算子执行网格传递，效率要高得多，从而使多层级训练在可行性和成本效益方面都大为提升。

我们首先注意到，对于两组节点 T， T'，其中 $T \subset T'$，有 $S_r(T') \subset S_r(T)$。因此，在嵌套网格的情况下，存在一个将一个网格映射到另一个网格的插值算子 $\mathcal{P}$。在任意粗粒度网格上，由任意一组节点 T 和基 $\left\{b_i^{[r]}(\cdot;T)\right\}_{i=1-r}^{n-1}$ 定义的样条函数，在更细网格上都对应一个节点为 T' 的基，该基由 $\left\{b_i^{[r]}(\cdot;T')\right\}_{i=1-r}^{m-1}$ 给出，其中 $m > n$。因此，对于节点为 T 的 KAN 层，我们可以定义一个节点为 T' 的细粒度 KAN 层，其值保持不变，因为对于插值矩阵 $\mathcal{P}$，有

$$\begin{aligned} x_q^{(\ell+1)} &= \sum_{p=1}^{P}\sum_{i=1-r}^{n-1} \widetilde{W}_{qpi,T}^{(\ell)}\, b_{i,T}^{[r]}(x_p^{(\ell)}) = \sum_{p=1}^{P}\sum_{i=1-r}^{n-1}\sum_{j=1-r}^{m-1} \widetilde{W}_{qpi,T}^{(\ell)}\, \mathcal{P}_{ij}\, b_{j,T'}^{[r]}(x_p^{(\ell)}) \\ &:= \sum_{p=1}^{P}\sum_{j=1-r}^{m-1} \widetilde{W}_{qpj,T'}^{(\ell)}\, b_{j,T'}^{[r]}(x_p^{(\ell)}). \end{aligned} \tag{24}$$

对于任意 $T \subset T'$ 的一般情况，使用由公式 (5) 定义的相应换基矩阵 $A_T^{[r]}$， $A_{T'}^{[r]}$，我们可以写成

$$\mathcal{P} = A_T^{[r]}\, \mathcal{I}\, {A_{T'}^{[r]}}^{-1}, \quad \text{where} \quad \mathcal{I}_{ij} = \begin{cases} 1 & t_i = t'_j \text{ for } t_i \in T,\, t'_j \in T' \\ 0 & \text{else} \end{cases}.$$

对于均匀样条网格上的二进细化， $\mathcal{P}$ 可明确表示为 [25] a s

$$\mathcal{P}_{ij} = \begin{cases} 2^{-r}\binom{r}{s} & j = 2i + s \\ 0 & \text{else} \end{cases}. \tag{25}$$

式 (24) 中描述的细化过程会生成一个新的 KAN 层，其函数与前一层相同，但拥有更多可训练参数。根据此定义，我们满足定义 1，因此可以定义一个恰当嵌套层次结构 (21)，而无需在每一层都评估具有细层级计算成本的网络。这是多层 KAN 的一项基本特性，因为这意味着细化过程不会改变粗层模型上已取得的训练进展。

迄今为止，KAN 的多层级方法并非 B 样条方法所独有。传递算子和恰当嵌套层次结构适用于广泛类别的 KAN，其中激活函数经过离散化。对于激活函数的任何离散化，只要在网格（或多项式）细化时空间具有类似的嵌套关系，就满足相同的结论，我们将在下文陈述这一结论。

引理 6. 任何激活函数定义在某个基上、且该基在细化过程中具有嵌套结构的 *KAN*，都可以配备插值算子，从而得到一个恰当嵌套层次结构（定义 *1*）。

证明。设 $\mathcal{B} = \{\phi_i \colon [a,b] \to \mathbb{R}\}_{i=1}^{n_B}$ 是定义域 $[a,b]$ 上激活函数空间的一组基，基的大小为 n_B （例如 B 样条、切比雪夫多项式、小波），其中 a， $b \in \mathbb{R}$, $a < b$；再设 $\mathcal{B}' = \{\phi_i' \colon [a,b] \to \mathbb{R}\}_{i=1}^{n_{B'}}$ 是一组大小为 $n_{B'} > n_B$ 的基，使得 $\mathcal{B}' \supset \mathcal{B}$。那么存在一个算子 $\mathcal{P}_{\mathcal{B}}^{\mathcal{B}'} \colon \mathcal{B} \to \mathcal{B}'$，它将基 $\mathcal{B}$ 插值到 $\mathcal{B}'$，使得对于 $\sigma \in \mathcal{B}$，有 $\mathcal{P}_{\mathcal{B}}^{\mathcal{B}'}\sigma \in \mathcal{B}'$ 且 $\sigma(x) \equiv \mathcal{P}_{\mathcal{B}}^{\mathcal{B}'}\sigma(x)$。 □

请注意，这种网络细化形式与激活函数的离散化有关；本质上，它是针对权重张量的通道维度进行细化，而不是增加每一层的隐藏节点数量。尽管已有一些通过增加层数或隐藏节点来细化网络的尝试，但引理 6 中的方法不会改变网络中的节点数量。

4.3 互补松弛

衰减不同误差模态的互补过程，可以说是成功的多重网格方法中最基本的组成部分，也是多层机器学习面临的最大挑战。此前我们讨论过，KAN 能够构建定义明确的机器学习模型嵌套层级。此外，不同层级上的“松弛”或局部训练必须具有互补性，这一点至关重要：当你将模型插值到更细层级时，优化过程应能利用这种新增的表达能力。

根据引理 4 和表 1 的结果，我们可以根据梯度几何和目标空间的几何，在基于下降的多通道多层感知机/KAN 优化中，强烈倾向于学习相对于样条节点而言频率较高或较低的函数。为简单起见，考虑梯度下降，$D_{\mathbf{u}}$， $D_{\mathbf{w}} = I$。回顾式 (17a)，在 ReLU 几何和基下进行优化，会在样条基中施加预条件。为保证稳定性，学习率必须设置为使预条件算子的最大特征模态归一化。当预条件器 $(AA^T)^{-1}$ 的谱范围很大时，与 $(AA^T)^{-1}$ 的小特征值相关的梯度几乎无法修正权重。如第 2.2 节所述，偶数阶微分算子的特征函数按从最光滑到最振荡的顺序排列，其幅值从小到大，而 $(AA^T)^{-1} \sim (\partial_x)^{-r}$ 则恰好相反。此外，特征值的幅值范围很广。对于 $r = 1$，我们有 $(AA^T)^{-1} \sim \Delta^{-1}$，其中最光滑的模态为 $\mathcal{O}(1)$ 阶，而样条网格上的振荡模态为 $\mathcal{O}(h)^2$ 阶，节点间距为 h。因此，采用基诱导预条件 $(AA^T)^{-1}$ 的基于梯度的优化，会使节点空间中光滑的梯度修正所受权重比振荡的梯度修正高出若干数量级。对于 $r > 1$，这一点尤其明显。因此，多通道多层感知机不满足互补松弛，因为所有 层都会强烈偏向于优化节点空间中的光滑函数。这与我们的目标恰恰相反，因为细化样条节点后，我们通过更复杂/更不光滑的函数增加了新的表达能力。如果优化器无法快速优化这个新空间，那么细化就毫无意义。我们将在第 5 节通过数值实验展示这一现象。

相比之下，在 KAN 自然的样条基中，待优化的自由权重对应于具有紧支撑的局部基函数。因此，某个特定权重上非零的梯度会产生更新，其影响主要局限于较大 KAN 函数复合中对应样条节点周围的局部区域。为此，我们预期梯度会直接传递到这些系数上，并自然支持分辨率与样条节点相当的振荡函数，这主要得益于紧支撑和局部化特性。这正是多层级训练场景所需的行为：当我们对样条节点进行均匀细化时，细化后模型上的基于梯度的优化器会立即开始利用新增的表达能力来降低损失。事实上，我们将在下一节通过数值实验准确展示这一行为。

5 数值结果

5.1 回归

首先，我们考虑对一个非光滑函数进行 0.175 弧度的逆时针坐标旋转（$\approx 10.03°$）后的函数回归

$$f(x,y) = \cos(4\pi x) + \sin(\pi y) + \sin(2\pi y) + |\sin(3\pi y^2)|, \tag{26}$$

采用 $[2,5,1]$ 架构。优化时，使用样条基（在第3节的语境下为$\mathbf{u}$ ）定义几何和梯度，并采用下降法（14b）；同时使用 ReLU 基（在第3节的语境下为$\mathbf{w}$ ）定义几何和梯度，并采用下降法（17a）。在这两种情况下，我们分别训练粗模型、细模型和采用多层级训练计划的模型，并与相应的普通多层感知机架构进行比较。我们使用 L-BFGS 进行训练，各模型在训练期间的工作量（浮点运算次数）相同。训练轮数以列表形式表示，例如 $\{32,16,8,4\}$，这意味着先在初始模型上训练 32 轮，然后转移到细化后的模型上训练 16 轮；细化时，原始网格均匀细分一次，后续步骤以此类推。*粗模型* 对应于$\{128,0,0,0\}$，*细模型* 对应于 $\{0,0,0,16\}$，而 *多级模型* 则对应于 $\{32,16,8,4\}$。结果见表 2，为五次随机初始化的平均值。

表 2：在不同基、架构和训练方案下训练的回归准确率。标准差基于随机初始化计算（$N=5$）。

Type	Layers	Basis	Fidelity	# Param.	MSE: mean (stdev)
KAN	$[2,5,1]$	ReLU	coarse	55	$1.10\times10^{-2}(8.32\times10^{-3})$
KAN	$[2,5,1]$	ReLU	fine	230	$1.35\times10^{-0}(1.50\times10^{-1})$
KAN	$[2,5,1]$	ReLU	multilevel	230	$1.06\times10^{-2}(8.03\times10^{-3})$
KAN	$[2,5,1]$	Spline	coarse	55	$1.65\times10^{-3}(4.18\times10^{-5})$
KAN	$[2,5,1]$	Spline	fine	230	$2.54\times10^{-3}(5.64\times10^{-3})$
KAN	$[2,5,1]$	Spline	multilevel	230	$\mathbf{3.67\times10^{-5}}(7.19\times10^{-5})$
MLP	$[2,5,1]$	ReLU		20	$2.94\times10^{-2}(1.60\times10^{-2})$
MLP	$[2,30,1]$	ReLU		120	$1.02\times10^{-3}(9.42\times10^{-4})$
MLP	$[2,20,20,1]$	ReLU		500	$3.33\times10^{-4}(2.92\times10^{-4})$

首先，我们看到，在样条基$\mathbf{u}$中进行训练，与ReLU基$\mathbf{w}$相比，准确率提升了 $1-3$ 个数量级。同样明显的是，在训练轮次固定的情况下，原则上表达能力更强的细模型，对于样条基至多只能达到与粗模型相当的准确率，而对于ReLU基则要差 两个数量级。在样条基中进行多层级训练可显著加速训练，其准确率比仅在粗模型或细模型上训练高出数个数量级，并且比规模更大的普通多层感知机高出一个数量级。相比之下，在ReLU基中进行多层级训练，相对于仅使用粗模型几乎没有任何改进。基于第3节的理论，这一点完全合理。当我们细化样条节点，并开始在ReLU基中对更细的模型进行基于梯度的训练时，相应的预条件会以数个数量级强烈偏向几何光滑模态。然而，几何光滑模态在很大程度上本来就可以由粗模型表示，这导致在训练细化后的

ReLU基中的模型时，准确率没有任何提升。这是多层级机器学习背景下的一项重要观察，实际上是将多重网格逼近性质扩展到随机优化的情境中——关键在于，在细模型上的训练必须与在粗模型上的训练形成互补。

图4展示了在相同训练工作量下的收敛历史，以说明细化对训练历史的影响。多级模型的损失停滞后突然下降的点，对应于模型进行细化的训练轮次。请注意，细模型（KAN 或多层感知机）的训练实际上已陷入停滞，损失下降极其缓慢。这凸显了多层级训练的价值：损失变化图表明，单独训练细粒度的 KAN 或多层感知机模型，即使最终能够达到多层级训练所取得的准确率，也需要显著更多的训练轮数。

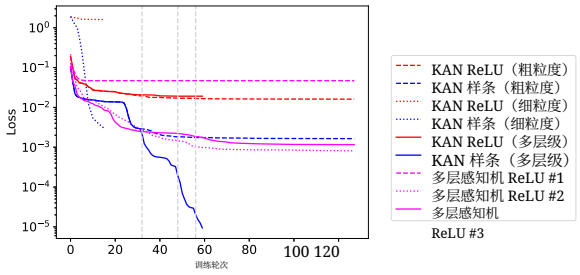


图4：不同模型在近似相同工作量下进行回归时的部分收敛历史。竖线表示多级模型的细化节点。

5.2 物理信息神经网络

本节探讨多层级训练物理信息神经网络（PINN）的影响。如今，PINN 已确立为逼近偏微分方程解的领先神经网络方法。训练这些模型可能颇具难度，需要运用多种技巧才能高效地获得合理的解 [64, 63]。我们的目标是使用 PINN，研究与传统架构相比，我们的 KAN 方法的性能特征。为此，我们限制使用能够产生先进物理信息神经网络解的专门数据修改方法和训练技术。其中一项包含的 增强方法是基于残差的注意力机制（RBA），该方法在 [3], 中被提出并应用于体积损失项。RBA 在不掩盖结果解读的情况下改善了收敛性。

5.2.1 二维泊松方程

第一个考虑的 PINN 用于求解二维泊松方程：

$$\begin{aligned} \nabla \cdot (\epsilon(\vec{x}) \nabla u(\vec{x})) = f(\vec{x}) \quad \vec{x} \in [-1,1]^2, \\ u(\vec{x}) = 0 \quad x \in \partial\Omega. \end{aligned} \tag{27}$$

令 $\vec{x} = [x, y]$，系数和制造解为

$$\epsilon(\vec{x}) = \begin{cases} \epsilon_l = 1 & x < 0 \\ \epsilon_r = \frac{1}{2} & x \geq 0 \end{cases}, \quad u^*(\vec{x}) = \begin{cases} \sin(\pi x)\sin(3\pi y) & x < 0 \\ \sin(2\pi x)\sin(3\pi y) & x \geq 0 \end{cases} \tag{28}$$

相应的源项为 $f(\vec{x})$。图 5（左）所示的 PINN 损失定义在三组等间距点上：体积 V（内部的 2401 个内部点）、边界 B（外部的 200 个点）以及界面 I（满足 $x = 0$ 的 49 个点）。$\mathcal{L}_I$ 项强制保证在 $x = 0$处梯度的正确跳跃。为了使用单个神经网络近似（而不是在每个子域中使用独立的网络），我们按照 [60]中的提议，为网络增添一个额外的水平集场。

图 5（右）展示了所研究的三种不同网络。两种 KAN 架构均采用多层级训练，最粗层包含 4 条样条。在随后的每个更细层级上，样条数量加倍。这些网络使用 Adam 进行训练。包括损失项加权在内的其他超参数可在补充材料中找到。损失和误差随训练轮次的变化如图 6 所示，针对

$$\mathcal{L}_V(\theta) = \sum_{\vec{x}\in V} \left|\nabla\cdot(\epsilon(\vec{x})\,\nabla u_N(\vec{x})) - f(\vec{x})\right|^2$$
$$\mathcal{L}_B(\theta) = \sum_{\vec{x}\in B} \left|u_N(\vec{x})\right|^2$$
$$\mathcal{L}_I(\theta) = \sum_{\vec{x}\in I} \left|(\epsilon_l \nabla u_N(\vec{x}_l) - \epsilon_r \nabla u_N(\vec{x}_r))\cdot[1,0]\right|^2$$
$$\mathcal{L}(\theta) = \gamma_V\,\mathcal{L}_V(\theta) + \gamma_B\,\mathcal{L}_B(\theta) + \gamma_I\,\mathcal{L}_I(\theta)$$

Type	Layers	Basis	# Param.
KAN	[2,5,1]	Spline	140, 220, 380, 700
KAN	[2,5,1]	ReLU	140, 220, 380, 700
MLP	[2,16,16,1]	ReLU	352

图 5：对于第 5.2.1 节中的二维泊松方程问题，左图展示了 PINN 损失中的体积、边界和界面分量；右图展示了多层级 KAN 和多层感知机架构及其参数数量。神经网络 u_N 假设依赖于参数 θ。

采用样条基的 KAN（左）、采用 ReLU基的 KAN（中）以及普通多层感知机（右）。浅灰色竖直虚线表示多层级算法切换层级时样条数量的增加。采用样条基的多层级 KAN 与普通多层感知机均取得了较小的损失和较小的误差。然而，采用 ReLU基的多层级训练未能取得进展，并停滞在略低于 $\mathcal{O}(1)$ 相对ℓ^2-误差处。注意，由于样条情形与 ReLU 情形构成相同的逼近空间，仅训练动态即可解释这一差异。我们推测，样条基的良好表现源于逐步学习更高的频率模态，正如第3节所发展的理论所表明的那样。

我们还看到了多层级训练带来的显著优势。采用样条基的多层级 KAN 相较于可比的多层感知机，能更快地达到较小的相对误差，并且实际误差随训练轮次变化时的波动显著更小。对于目标解未知的情况，后一点对于鲁棒性和泛化尤其重要。回到多层级 KAN 的训练，我们还要强调多层级训练中明显的阶梯效应。每次细化后，特别是在 1250 和 2500 个训练轮数时进行细化后，损失和误差在细化前已基本趋于平稳；细化后，损失和误差立即迅速下降，这表明模型能立即利用其表达能力的提升。这种行为体现了多层层次和训练方法的成功，也是在研究机器学习多层方法的文献中，我们尚未见过的现象。

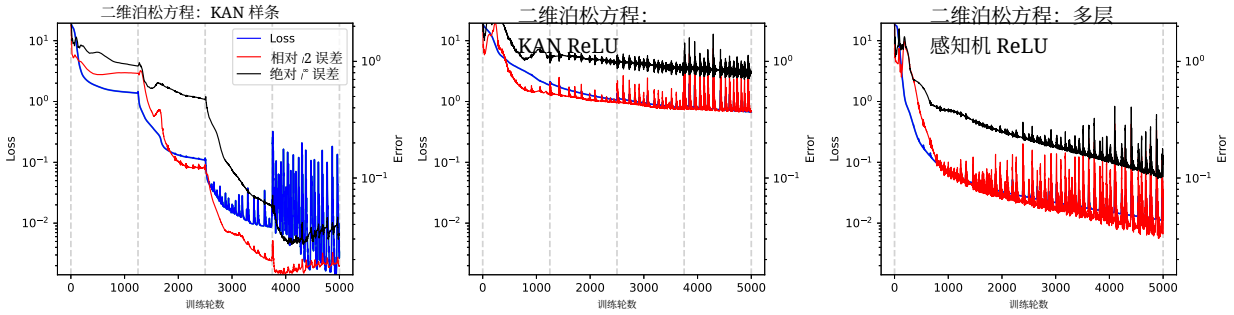


图 6：二维泊松方程：采用样条基进行 KAN 多层级训练（左）、采用 ReLU 基进行 KAN 多层级训练（中），以及训练可比的多层感知机（右）时，损失和误差随训练轮次变化的曲线。KAN 样条与 KAN ReLU 的逼近能力相同，但多层级训练性能的差异表明，KAN 样条的松弛与细化策略相辅相成。

5.2.2 一维 Burgers 方程

接下来，我们来看一个简短示例：训练物理信息神经网络来求解一维 Burgers 方程 $u_t + uu_x - \nu u_{xx} = 0$，其中 $\nu = 10^{-2}$是一个正则性本质上较低的物理问题。解表示在 64×64 网格上的时空配点处，并在每个优化步骤中使用整个批次。我们采用权重正则化 Adam，并像其他地方（例如[64]）那样，额外使用指数循环学习率调度器。基函数结果运行 10000 个步骤；进行细化时，粗模型使用 $[3200, 0, 0]$ 个训练轮次

，细模型使用 $[0, 0, 800]$ 个训练轮次（比回归问题的层级低一级），多级模型使用 $[800, 400, 200]$ 个训练轮次。鉴于此前展示的 ReLU 基表现不佳，这个问题中我们只展示使用样条基的结果。结果见表 3。KAN架构——即使是最粗糙的架构——在训练损失方面也优于可比的多层感知机架构。多层级训练带来了显著提升：相较于在细粒度或粗粒度架构下单独使用 KAN 或多层感知机，在工作量相近的情况下，准确率提高了 2–3 个数量级。

表 3：物理信息神经网络问题的准确率，比较样条节点的几何细化对 KAN 基的影响。表中还列出了具有可比性的多层感知机。标准差根据随机初始化计算（$N = 5$）。

Type	Layers	Fidelity	# Param.	Loss: mean (stdev)
KAN	$[2, 20, 20, 1]$	coarse	3400	$6.635 \times 10^{-3} (4.060 \times 10^{-3})$
KAN	$[2, 20, 20, 1]$	fine	9700	$4.072 \times 10^{-3} (2.514 \times 10^{-3})$
KAN	$[2, 20, 20, 1]$	multilevel	9700	$\mathbf{2.402 \times 10^{-5}} (1.897 \times 10^{-5})$
MLP	$[2, 20, 20, 1]$		500	$2.334 \times 10^{-2} (1.740 \times 10^{-3})$
MLP	$[2, 56, 56, 1]$		3416	$1.734 \times 10^{-2} (7.989 \times 10^{-3})$

5.2.3 Allen-Cahn方程

最后，我们考虑一个物理信息神经网络问题，学习 Allen-Cahn方程的解：

$$\begin{aligned} \partial_t u(x,t) &= -\epsilon \partial_{xx} u(x,t) + 5u(x,t)^3 - 5u(x,t) && \text{for } (x,\ t) \in [-1,1] \times [0,1] \\ u(x,0) &= x^2 \cos(\pi x) && \text{for } (x,t) \in [-1,1] \times \{0\}. \end{aligned}$$

我们借此再次展示多层级训练的强大之处，以及多层级训练所带来的模型输出频谱整形效果。

随着时间增加，该方程在 $u = \pm 1$ 处存在稳定吸引子，在 $u = 0$ 处存在一个不稳定不动点；尽管后者能最小化偏微分方程的残差，却会产生与初始条件不相容的解。与前面的例子一样，我们在训练过程中使用基于残差的注意力机制，但除此之外，不使用任何当前最先进解法中常见的专门修改和训练技术，包括但不限于时间因果性、更先进的优化器，以及对配点进行自适应采样。这里要强调的是，KAN 用于物理信息神经网络的多层级训练开箱即用，快速而稳健，无需任何修改。我们使用 Adam优化器训练每个模型，在每种分辨率下训练 2500 步，学习率为 0。001；模型架构采用 sigmoid归一化层和三次B样条，起始分辨率为 3 个内部样条节点，均匀分布在区间 $[-1, 1]$ 上。训练过程中，在定义域内随机选取 20，000 个配点来评估损失中的残差。

图 7 展示了采用两种基以及规模相近的多层感知机时的训练损失变化曲线，以及所得的物理解。与前文一样，我们看到，样条基中的多层级训练带来了显著优势，包括训练立即加速，以及每个细化层级的损失均有所降低。相比之下，ReLU基无法利用细化所增加的网络容量，导致解的质量很差。此外，规模相近的传统多层感知机表现与 ReLU KAN 相似，无法捕捉解中的任何有意义结构。

我们进一步考察了图 8 中基于样条的 KAN 进行多层级训练时产生的残差和频谱（基于 ReLU 的 KAN 的类似图表见补充材料）。值得注意的是，我们清楚地看到，使用样条基时，细化不仅能提高准确率，还会显著改变残差中傅里叶模态的频谱：随着细化，傅里叶模态的能量降低，而随着样条表示不断细化，残差中活跃的傅里叶模态的支撑范围会扩大。这些结果进一步印证了前几节理论得出的结论：在样条基中进行分层训练，能够通过细化有针对性地改善高阶傅里叶模态。相比之下，ReLU 网络在样条分辨率提高时虽有所改善，但其频谱仍然狭窄——正如谱偏差所表明的，多层级训练并非发生在我们能够利用多重网格式细化的几何空间中——因此模型的准确率会受到影响，尽管两种架构具有相同的逼近能力，且属于同一类别。

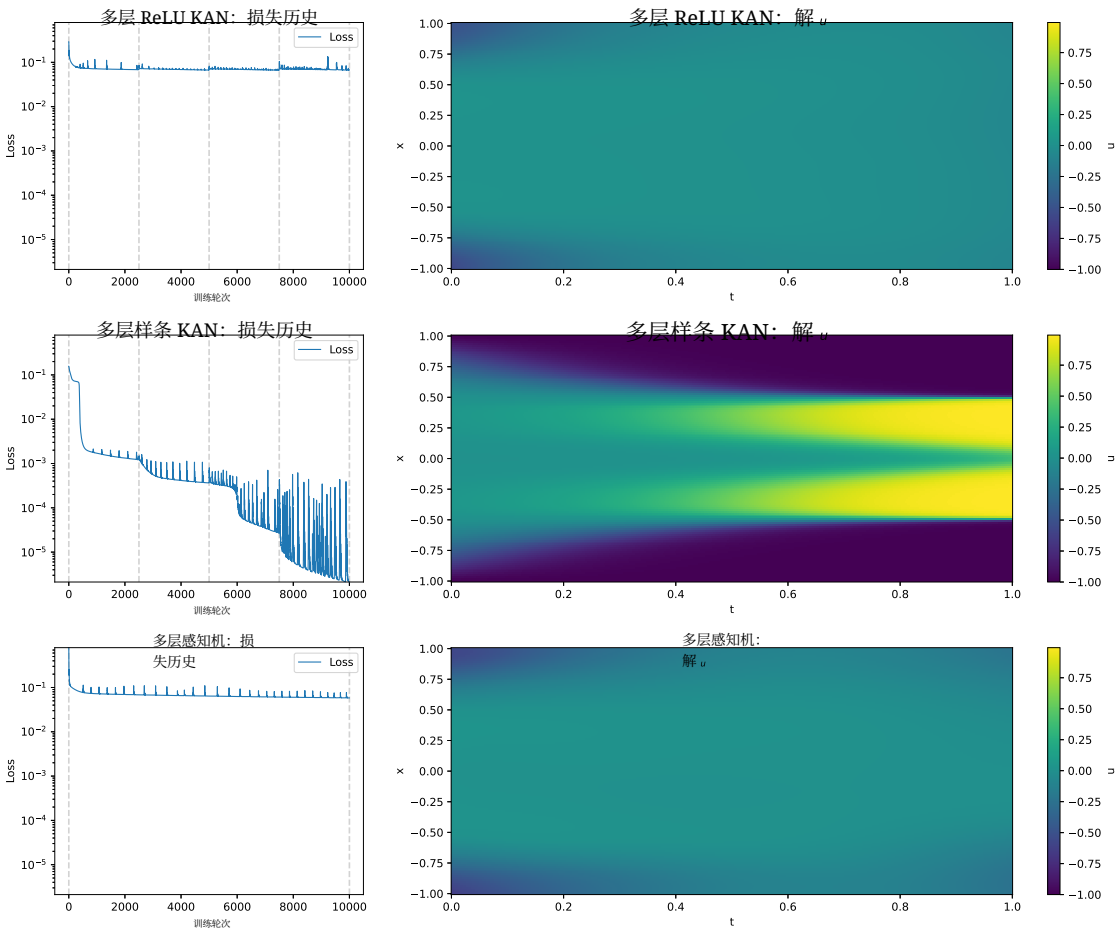


图 7：ReLU 和 B 样条基下等效 KAN 架构进行多层次细化的训练历史，并与规模相近的多层感知机进行比较。

6 结论

我们提出了一个理论与算法框架，运用多层次原理理解并高效训练基于样条的 Kolmogorov-Arnold 网络。从理论角度看，我们证明，具有阶数为 r 的样条基函数的 KAN，通过特定的线性基变换，与采用幂 ReLU 激活函数的多通道多层感知机等价，其中变换矩阵 $A^{[r]}$ 对应于 r阶导数的有限差分离散化。这种等价性带来了直接的算法优势：采用非递归实现的基于样条的 KAN，比标准 Cox-de Boor 实现更快。更重要的是，我们的分析揭示了一个根本差异：尽管不同基表示作为前向算子等价，基于梯度的优化在这些表示下的行为却截然不同。基变换矩阵 $A^{[r]}$ 充当预条件器，会显著影响训练动态。具体而言，在多通道多层感知机（ReLU）基中进行梯度下降，会以数量级的缩放差异，极度优先拟合光滑函数而非振荡函数。相比之下，自然样条基凭借紧支撑实现了强大的特征定位，使网络能够高效学习具有尖锐梯度和低正则性的函数。这一理论认识为 KAN 在捕捉复杂非光滑函数方面所表现出的优势提供了严格的理论依据。

基于这些见解，我们开发了一种基于均匀节点细化的高效多层级训练框架。我们引入了用于多层次优化的适当嵌套的层次结构这一概念，确保粗模型能够保留在更细层级上实现的函数近似——这是防止插值模型抵消已有进展的关键属性，

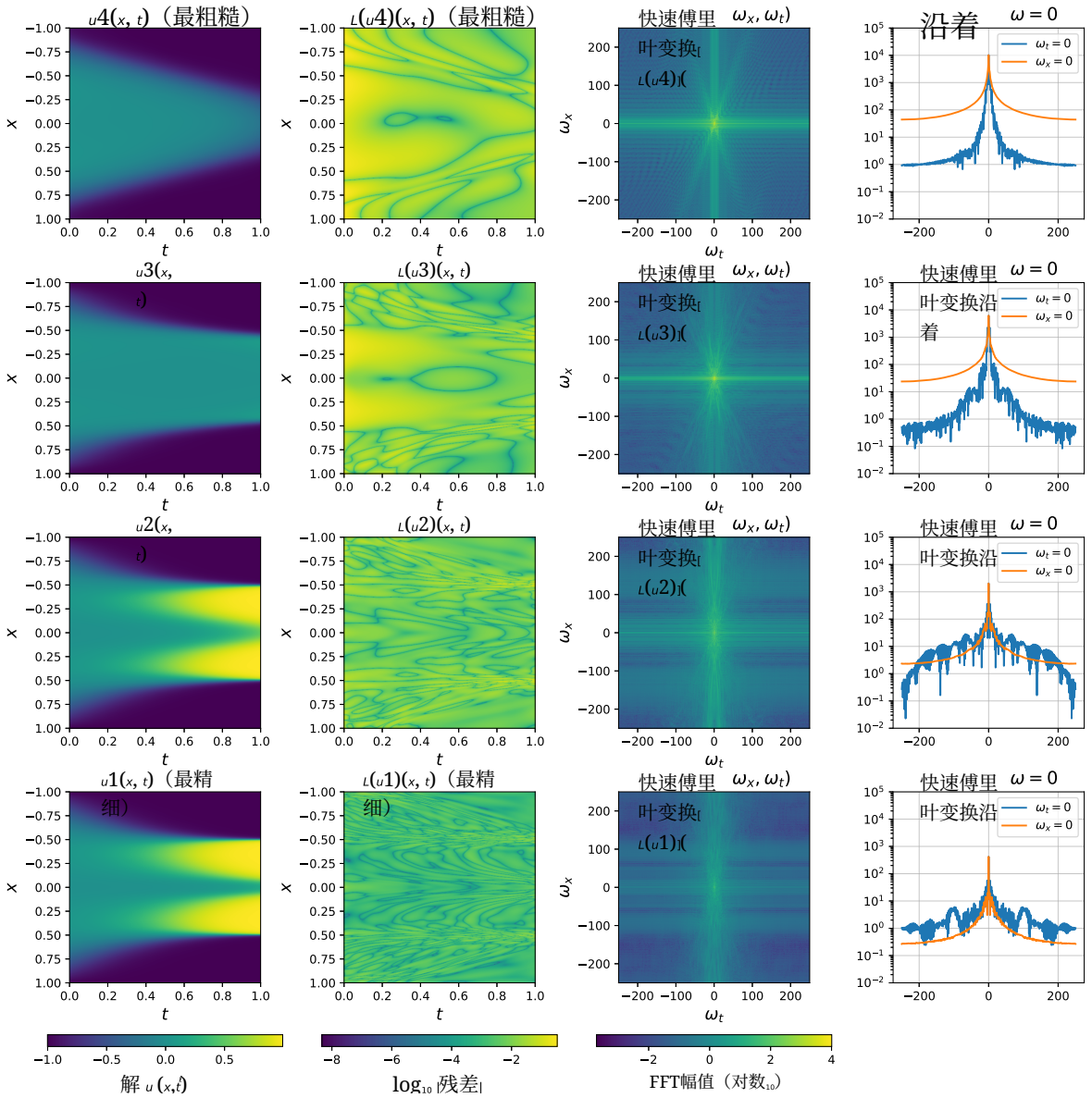


图 8：第 5.2.3 节中 Allen-Cahn方程的训练后的B样条基解（最左列）、对数尺度下的残差（左中列）、残差的傅里叶变换（右中列），以及沿坐标轴 $\omega_t = 0$ 和 $\omega_x = 0$ 的傅里叶变换截面（最右列）。每行显示从最粗糙（顶部）到最精细（底部）各细化层级训练后的输出。前三列共用同一色条，显示在每列底部。

当该框架与自然样条基中的基于梯度的下降方法相结合时，自然样条基会自动在各层级之间提供互补松弛，从而实现了显著改进：我们的数值实验一致表明，与标准训练方法或同类多层感知机相比，准确率提高了 2–3 个数量级，同时保持了计算效率。相比之下，我们还表明，尝试使用 ReLU基表述进行多层级训练，相较于仅使用粗模型实际上没有任何改进，这证实成功的多层方法需要适当嵌套以及跨层级的互补优化动力学。这一原则在多层级机器学习中一直难以实现，但在KAN的原生样条表示中却能自然满足。

更广泛地说，我们的多层级框架有力地表明，只要结构设计得当，

并在不同层级间形成互补的动态机制，多重网格思想确实能够提升神经网络训练的效率，正如它们长期以来在科学计算中所发挥的作用。未来的工作将探讨多层循环，以及将该方法扩展到其他架构和不同类型的粗化与细化。

多层感知机架构的记号变换

通常，一个具有 L 个隐藏层、以输入 $x = x^{(0)}$ 开始的多层感知机，其每一层 $\ell = 0, \ldots, L-1$ 都表示为 $x^{(\ell+1)} = \sigma\left(W_{(\ell)}x^{(\ell)} + b^{(\ell)}\right)$，最终输出 $NN(x) = x^{(L+1)} = W_{(L)}x^{(\ell)}$.这种组合结构可以进一步细致地分解为：

$$x^{(\ell+\frac{1}{2})} = W^{(\ell)}x^{(\ell)}, \qquad x^{(\ell+1)} = \sigma\left(x^{(\ell+\frac{1}{2})} + b^{(\ell)}\right), \tag{29}$$

其中，最后一层省略非线性激活步骤，因此 $NN(x) = x^{(L+\frac{1}{2})}$而不是 $NN(x) = x^{(L+1)}$.

在我们的分析中使用的重排层，将各层与可学习权重相乘（而非用可学习权重预乘），只需将式 (29) 中 $\ell = 1, \ldots, L-1$ 的表达式合并为一个步骤，$x^{(\ell+\frac{1}{2})} = W_{(\ell)}\sigma\left(x^{(\ell-\frac{1}{2})} + b^{(\ell-1)}\right)$，即可得到相同的计算结果。在记号上，我们将$t^{(\ell)} = -b^{(\ell)}$替换为相应记法，以符合样条文献中的惯例。由此得到每一层的最终表达式：

$$x^{(\ell+\frac{1}{2})} = W^{(\ell)}\sigma\left(x^{(\ell-\frac{1}{2})} - t^{(\ell-1)}\right). \tag{30}$$

B 关于基变换矩阵 $A^{[r]}$的证明

首先，我们证明以下关于ReLU函数的结果。

命题2. 对于任意标量 a、b，以及对于 $r > 1$,

$$(x-a)ReLU(x-b)^r = ReLU(x-b)^{r+1} + (b-a)ReLU(x-b)^r. \tag{31}$$

证明。 若 $x \le b$，则两边均等于零。因此，只需考虑 $x > b$，此时ReLU$(x-b) = x-b$.因此，只需证明$(x-a)(x-b)^r = (x-b)^{r+1} + (b-a)(x-b)^r$. 事实上，对右端进行因式分解，我们看到

$$(x-b)^{r+1} + (b-a)(x-b)^r = (x-b)^r\left((x-b) + (b-a)\right) = (x-b)^r(x-a), \tag{32}$$

此即所需结果。 □

现在我们推导 $A^{[r]}$的递归定义，如(5)所述。

引理7 (引理 2)。 对于 $r > 1$，令 $A^{[r-1]}$ 表示在阶为 r 的样条的节点上构造的阶为 $r-1$ 的样条。则矩阵 $A^{[r]}$ 按元素定义为：

$$A_{ij}^{[r]} = \frac{1}{t_{i+r-1} - t_i}A_{ij}^{[r-1]} - \frac{1}{t_{i+r} - t_{i+1}}A_{i+1,j}^{[r-1]}, \tag{33}$$

其中我们定义 $A_{n+r}^{[r-1]} := 0$，以处理最后一行 $i = n + r - 1$ 的特殊情况。

证明。 容易验证

$$A_{ij}^{[1]} = \begin{cases} 1 & j = i \\ -1 & j = i+1 \end{cases}. \tag{34}$$

我们现在用归纳法证明主要命题。对于 $r=2$ 的基例，我们有已知的（例如 [26]）结果，即

$$A_{ij}^{[2]}=\begin{cases}\frac{1}{t_{i+1}-t_i} & j=i\\ -\left(\frac{1}{t_{i+1}-t_i}+\frac{1}{t_{i+2}-t_{i+1}}\right) & j=i+1\\ \frac{1}{t_{i+2}-t_{i+1}} & j=i+2\\ 0 & \text{else}\end{cases}. \tag{35}$$

由（34）和（35）可直接看出， $A^{[2]}$ 满足关于 $r=2$的方程（33）。现在我们继续进行关于 $r\geq 2$的归纳步骤。假设结论对于 $s=1$、.....、 $r-1$均成立。根据用于定义 $b_i^{[r]}$ 的递推关系，我们有

$$\begin{aligned}b_i^{[r]}(x)&=\frac{x-t_i}{t_{i+r-1}-t_i}b_i^{[r-1]}(x)+\frac{t_{i+r}-x}{t_{i+r}-t_{i+1}}b_{i+1}^{[r-1]}(x)\\&=\left(\frac{1}{t_{i+r-1}-t_i}\right)\sum_j A_{ij}^{[r-1]}\,(x-t_i)\,\mathrm{ReLU}(x-t_j)^{r-2}-\\&\qquad\left(\frac{1}{t_{i+r}-t_{i+1}}\right)\sum_j A_{i+1,j}^{[r-1]}\,(x-t_{i+r})\,\mathrm{ReLU}(x-t_j)^{r-2}.\end{aligned}$$

根据引理 2,

$$\begin{aligned}b_i^{[r]}(x)&=\left(\frac{1}{t_{i+r-1}-t_i}\right)\sum_j A_{ij}^{[r-1]}\,(x-t_i)\,\mathrm{ReLU}(x-t_j)^{r-2}-\\&\qquad\left(\frac{1}{t_{i+r}-t_{i+1}}\right)\sum_j A_{i+1,j}^{[r-1]}\,(x-t_{i+r})\,\mathrm{ReLU}(x-t_j)^{r-2}\\&=\left(\frac{1}{t_{i+r-1}-t_i}\right)\sum_j A_{ij}^{[r-1]}\left(\mathrm{ReLU}(x-t_j)^{r-1}+(t_j-t_i)\mathrm{ReLU}(x-t_j)^{r-2}\right)-\\&\left(\frac{1}{t_{i+r}-t_{i+1}}\right)\sum_j A_{i+1,j}^{[r-1]}\left(\mathrm{ReLU}(x-t_j)^{r-1}+(t_j-t_{i+r})\mathrm{ReLU}(x-t_j)^{r-2}\right).\end{aligned}$$

由于 $b_i^{[r]}\in C^{r-2}([a,b])$ 和 $\mathbf{ReLU}(\cdot-t_j)^{r-1}\in C^{r-2}([a,b])$，而 $\mathbf{ReLU}(\cdot-t_j)^{r-2}\notin C^{r-2}([a,b])$，因此 $\mathbf{ReLU}(\cdot-t_j)^{r-2}$ 项的系数必须相互抵消。具体而言，将 $\mathbf{ReLU}(\cdot-t_j)^{r-2}$ 项展开为分段多项式时，展开后的各项在节点处都不够光滑，因此，为使整个求和结果属于 $C^{r-2}([a,b])$，所有这些项都必须相互抵消。通过繁琐的算术计算也可以直接证明这一点。因此,

$$\begin{aligned}b_i^{[r]}(x)&=\left(\frac{1}{t_{i+r-1}-t_i}\right)\sum_j A_{ij}^{[r-1]}\mathrm{ReLU}(x-t_j)^{r-1}-\\&\qquad\left(\frac{1}{t_{i+r}-t_{i+1}}\right)\sum_j A_{i+1,j}^{[r-1]}\mathrm{ReLU}(x-t_j)^{r-1}\\&=\sum_j\left(\frac{1}{t_{i+r-1}-t_i}A_{ij}^{[r-1]}-\frac{1}{t_{i+r}-t_{i+1}}A_{i+1,j}^{[r-1]}\right)\mathrm{ReLU}(x-t_j)^{r-1},\end{aligned}$$

证明至此完成。 □

我们现在推导 $A^{[r]}$ 在均匀节点下的闭式。对于 $r=2$，这是 [26]、 $A=I-2S+S^2$中对应于移位算子 $S\mathbf{e}_j=\mathbf{e}_{j+1}$的矩阵。

推论 1 (推论 1)。 在均匀节点的情况下，间距为 $h=t_i-t_{i-1}=\frac{1}{n}$， $A^{[r]}$ 可直接表示为 –

$$A^{[r]}=\frac{h^{1-r}}{(r-1)!}(A^{[1]})^r. \tag{36}$$

此外， $A^{[r]}$ 是上三角托普利兹矩阵，其元素为

$$A_{ij}^{[r]} = \begin{cases} \frac{(-1)^{j-i}}{(j-i)!\,(r-j+i)!} \frac{r}{h^{r-1}} & i \le j \le i+r \\ 0 & else \end{cases}. \tag{37}$$

证明。这可由如下观察得出：若 $t_{i+r-1} - t_i = t_{i+r} - t_{i+1} = h(r-1)$，则 (5) 对应于用 $h^{\frac{1}{(r-1)}} A^{[1]}$ (34) 对 $A^{[r-1]}$ 进行左缩放。对 $r = 2, 3, \ldots$ 重复此过程即得 (36)。关于托普利兹性质，由 (36) 可见 $A^{[r]}$ 是 $A^{[1]}$ 的幂。由于 $A^{[1]}$ 是上双对角托普利兹矩阵，因此 $A^{[r]}$ 也是上三角托普利兹（参见例如 [55,引理 22]），其系数由 (37) 给出。□

致谢

本工作部分由国家核安全管理局实验室间实验室主导研究与开发计划资助，项目编号为 20250861ER。洛斯阿拉莫斯国家实验室报告编号为 LA-UR-26-21552。本文由桑迪亚国家技术与工程解决方案有限责任公司的员工根据与美国能源部（DOE）签订的合同编号 DE-NA0003525 撰写。员工拥有本文的全部权利、所有权和权益，并对其内容负全部责任。美国政府保留相关权利，且出版商通过接受本文发表，承认美国政府保留一项非独占、已付清、不可撤销的全球许可，可为美国政府目的出版或复制本文的发表形式，或允许他人这样做。能源部将根据能源部公共访问计划 https://www.energy.gov/downloads/ doe-public-access-plan 向公众提供这些联邦资助研究的成果。BSS 感谢 Achi Brandt 和 Oren Livne 就机器学习的多层方法进行的多次早期且富有启发的讨论。

参考文献

[1] D. W. Abueidda, P. Pantidis, and M. E. Mobasher,*Deepokan*：基于 *Kolmogorov-Arnold* 网络的力学问题深度算子网络, arXiv 预印本 arXiv:2405.19143, (2024).

[2] J. Actor，《柯尔莫哥洛夫叠加定理的计算》，硕士论文，莱斯大学，*2018*。

[3] S. J. Anagnostopoulos, J. D. Toscano, N. Stergiopulos, and G. E. Karniadakis，《物理信息神经网络中的残差型注意力》，《计算机方法在应用力学与工程中的应用》，*421*（*2024*），第 *116805* 页，*https://doi.org/https://doi.org/10.1016/j.cma.2024.116805*，*https://www.sciencedirect.com/science/article/pii/S0045782524000616*。[4] C. Botsaris，微分梯度方法，《数学分析与应用杂志》，*63*（*1978*），第 *177–198* 页，*https://doi.org/10.1016/0022-247x(78)90114-2*。[5] J. Braun and M. Griebel，关于柯尔莫哥洛夫叠加定理的构造性证明，《构造性逼近》，*30*（*2009*），第 *653–675* 页。[6] R. Bresson, G. Nikolentzos, G. Panagopoulos, M. Chatzianastasis, J. Pang, and M. Vazir-giannis, *Kagnns*：*Kolmogorov-Arnold* 网络与图学习相结合，*arXiv* 预印本 *arXiv:2406.18380*，（*2024*）。[7] A. Cacciatore, V. Morelli, F. Paganica, E. Frontoni, L. Migliorelli, and D. Berardini, 使用 *Kolmogorov-Arnold* 网络进行计算机视觉持续学习的初步研究，*arXiv* 预印本 *arXiv:2409.13550*，（*2024*）。[8] Y. Cang, L. Shi, et al., *KAN* 能否奏效？探索 *Kolmogorov-Arnold* 网络在计算机视觉中的潜力，*arXiv* 预印本 *arXiv:2411.06727*，（*2024*）。[9] B. Chang, L. Meng, E. Haber, F. Tung, and D. Begert，从动力系统视角看多层残差网络，载于国际学习表征会议，*2018*。

[10] R. T. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud，神经常微分方程，《神经信息处理系统进展》，*31*（*2018*）。[11] M. Cheon，展示 *Kolmogorov-Arnold* 网络在视觉任务中的效能，*arXiv* 预印本 *arXiv:2406.14916*，（*2024*）。[12] C. K. Chui，多元样条，工业与应用数学学会，*1988*。[13] E. C. Cyr, J. Hahne, N. S. Moore, J. B. Schroder, B. S. Southworth, and D. A. Vargas，*Torchbraid*：基于消息传递接口和 *GPU* 加速的深度神经网络高性能层并行训练，*ACM* 数学软件汇刊，*51*（*2025*），第 *1–30* 页。[14] C. De Boor，《样条函数实用指南》，第 *27* 卷，*Springer* 纽约，*1978*。[15] C. Dong, L. Zheng, and W. Chen，用于时间序列分类和鲁棒分析的 *Kolmogorov-Arnold* 网络（*KAN*），*arXiv* 预印本 *arXiv:2408.07314*，（*2024*）。[16] M. Eliasof, J. Ephrath, L. Ruthotto, and E. Treister，*MGIC*：通道内多重网格神经网络架构，*SIAM* 科学计算杂志，*45*（*2023*），第 *S307–S328* 页，*https://doi. org/10.1137/21m1430194*。[17] M. Feischl, A. Rieder, and F. Zehetgruber，迈向神经网络的最优分层训练，*arXiv*，（*2024*），*https://doi.org/10.48550/arxiv.2407.02242*，*https://arxiv.org/abs/ 2407.02242*。[18] L. Gaedke-Merzhäuser, A. Kopaničáková, and R. Krause，深度残差网络的多层最小化，*ESAIM*：论文集与综述，*71*（*2021*），第 *131–144* 页，*https://doi.org/10. 1051/proc/202171131*。[19] Y. Gao and V. Y. Tan，关于 *Kolmogorov-Arnold* 网络中（随机）梯度下降的收敛性，*arXiv* 预印本 *arXiv:2410.08041*，（*2024*）。[20] S. Günther, L. Ruthotto, J. B. Schroder, E. C. Cyr, and N. R. Gauger，深度残差神经网络的层并行训练，*SIAM* 数据科学数学杂志，*2*（*2020*），第 *1–23* 页，*https://doi.org/10.1137/19m1247620*。[21] E. Haber and L. Ruthotto，深度神经网络的稳定架构，《逆问题》，*34*（*2017*），第 *014004* 页。[22] W. Hackbusch，《多重网格方法及其应用》，第 *4* 卷，*Springer Science & Business Media*，*2013*。[23] J. He and J. Xu，*MgNet*：多重网格与卷积神经网络的统一框架，《中国科学：数学》，*62*（*2019*），第 *1331–1354* 页，*https://doi.org/10.1007/s11425-019-9547-2*，*https: //arxiv.org/abs/1901.10415*。[24] Y. He, Y. Xie, Z. Yuan, and L. Sun，*MLP-KAN*：统一深度表征与函数学习，*arXiv* 预印本 *arXiv:2410.03027*，（*2024*）。[25] K. Höllig，基于 *B* 样条的有限元方法，工业与应用数学学会，*2003*。[26] Q. Hong, J. W. Siegel, Q. Tan, and J. Xu，关于神经网络谱偏差对激活函数的依赖性，*arXiv* 预印本 *arXiv:2208.04924*，（*2022*）。[27] Y. Hou and D. Zhang，*Kolmogorov-Arnold* 网络（*KAN*）综合综述，*arXiv* 预印本 *arXiv:2407.11075*，（*2024*）。[28] A. A. Howard, B. Jacob, S. H. Murphy, A. Heinlein, and P. Stinis，有限基 *Kolmogorov-Arnold* 网络：面向数据驱动和物理信息问题的区域分解，*arXiv* 预印本 *arXiv:2406.19662*，（*2024*）。

[29] B. Igelnik and N. Parikh，*Kolmogorov*样条网络，《*IEEE*神经网络汇刊》，*14*（*2003*），第 *725–733* 页。[30] B. Jacob A. A. Howard and P. Stinis，*Spikans*：可分离物理信息*Kolmogorov–Arnold*网络，*arXiv* 预印本 *arXiv:2411.06286*，（*2024*）。[31] A. Jacot F. Gabriel and C. Hongler，神经切线核：神经网络中的收敛与泛化，《神经信息处理系统进展》，*31*（*2018*）。[32] T.-W. Ke M. Maire and S. X. Yu，多重网格神经架构，载于 *IEEE*计算机视觉与模式识别会议（*CVPR*）论文集，*2017* 年，第 *6665–6673* 页。[33] M. Kiamari M. Kiamari and B. Krishnamachari，*Gkan*：图*Kolmogorov–Arnold*网络，*arXiv* 预印本 *arXiv:2406.06470*，（*2024*）。[34] D. P. Kingma M. Welling et al.，自编码变分贝叶斯，*2013*。[35] B. C. Koenig S. Kim and S. Deng，*Kan-odes*：用于学习动力系统和隐含物理的 *Kolmogorov–Arnold* 网络常微分方程，《计算机在应用力学与工程中的方法》，*432*（*2024*），第 *117397* 页。[36] T. G. Kolda and B. W. Bader，张量分解及其应用，《*SIAM*评论》，*51*（*2009*），第 *455–500* 页。[37] A. N. Kolmogorov，关于通过单变量连续函数的叠加与加法表示多变量连续函数，载于《科学院院报》，第 *114* 卷第 *5* 期，俄罗斯科学院，*1957* 年，第 *953–956* 页。[38] A. Kopaničáková and R. Krause，深度残差网络的全局收敛多层级训练，《*SIAM* 科学计算杂志》，*0*（*2022*），第 *S254–S280* 页，*https://doi.org/10.1137/ 21m1434076*。[39] P.-E. Leni Y. D. Fougerolle and F. Truchetet，用于图像处理的 *Kolmogorov*样条网络，载于《图像处理：概念、方法、工具与应用》，*IGI Global*，*2013* 年，第 *54–78* 页。[40] Z. Liu Y. Wang S. Vaidya F. Ruehle J. Halverson M. Soljačić T. Y. Hou andM. Tegmark，*KAN*：*Kolmogorov-Arnold* 网络，*arXiv* 预印本 *arXiv:2404.19756*，（*2024*）。[41] G. Lorentz，度量熵、宽度与函数叠加，《美国数学月刊》，*69*（*1962*），第 *469–485* 页。[42] W. S. McCulloch and W. Pitts，神经活动中内在观念的逻辑演算，《数学生物物理学通报》，*5*（*1943*），第 *115–133* 页。[43] A. Noorizadegan S. Wang and L. Ling，*Kolmogorov–Arnold*网络实践指南，*arXiv* 预印本 *arXiv:2510.25781*，（*2025*）。[44] S. Patra S. Panda B. K. Parida M. Arya K. Jacobs D. I. Bondar and A. Sen，通过 *Efficient-KAN* 和 *Wav-KAN* 进行动力学分析的物理信息*Kolmogorov–Arnold*神经网络，*arXiv* 预印本 *arXiv:2407.18373*，（*2024*）。[45] A. Pinkus，神经网络中多层感知机模型的逼近理论，《数值学报》，*8*（*1999*），第 *143–195* 页。[46] Q. Qiu T. Zhu H. Gong L. Chen and H. Ning，*Relu-kan*：只需矩阵加法、点乘和 *ReLU* 的新型 *Kolmogorov-Arnold* 网络，*arXiv* 预印本 *arXiv:2406.02075*，（*2024*）。[47] R. Qiu Y. Miao S. Wang Y. Zhu L. Yu and X.-S. Gao，*Powermlp*：一种高效的 *KAN* 版本，载于 *AAAI*人工智能会议论文集，第 *39* 卷第 *19* 期，*2025* 年，第 *20069–20076* 页。

[48] S. Rigas M. Papachristou T. Papadopoulos F. Anagnostopoulos and G. Alexandridis, 网格依赖的物理信息 *Kolmogorov-Arnold* 网络的自适应训练，*IEEE Access*，（*2024*）。
[49] F. Rosenblatt，感知机：一种用于大脑中信息存储与组织的概率模型，《心理学评论》，*65*（*1958*），第 *386* 页。[50] L. Ruthotto and E. Haber，受偏微分方程启发的深度神经网络，《数学成像与视觉》，*62*（*2020*），第 *352–364* 页。[51] K. Shukla J. D. Toscano Z. Wang Z. Zou and G. E. Karniadakis，针对微分方程与算子网络的多层感知机和 *KAN* 表示的全面、公平比较，*arXiv* 预印本 *arXiv:2406.02917*，（*2024*）。[52] N. J. A. Sloane and S. Plouffe，《整数序列百科全书》，（无标题），（*1995*）。[53] C. C. So and S. P. Yung，用于更准确、稳健且快速求解物理信息神经网络（*PINN*）的高阶 *ReLU-KAN*（*HRKAN*），*arXiv* 预印本 *arXiv:2409.14248*，（*2024*）。
[54] S. Somvanshi S. A. Javed M. M. Islam D. Pandit and S. Das，*Kolmogorov-Arnold* 网络综述，*arXiv* 预印本 *arXiv:2411.06078*，（*2024*）。[55] B. S. Southworth，*Parareal* 和时间多重网格缩减的必要条件与紧致两层收敛界，《*SIAM* 矩阵分析与应用杂志》，*40*（*2019*），第 *564–608* 页。
[56] D. A. Sprecher，*Kolmogorov* 叠加定理的一个通用映射，《神经网络》，*6*（*1993*），第 *1089–1094* 页。[57] D. A. Sprecher，《从代数到计算算法：*Kolmogorov* 与希尔伯特第 *13* 问题》，*Docent Press*，*2017*。[58] P. Tilli，非厄米块托普利茨矩阵的奇异值与特征值，《线性代数及其应用》，*272*（*1998*），第 *59–89* 页。[59] J. D. Toscano V. Oommen A. J. Varghese Z. Zou N. A. Daryakenari C. Wu and G. E.Karniadakis，从 *PINN* 到 *PIKAN*：物理信息机器学习的最新进展，*arXiv* 预印本 *arXiv:2410.13228*，（*2024*）。[60] Y.-H. Tseng T.-S. Lin W.-F. Hu and M.-C. Lai，一种用于椭圆界面问题的尖点捕获物理信息神经网络，《计算物理学杂志》，*491*（*2023*），第 *112359* 页。[61] C. J. Vaca-Rubio L. Blanco R. Pereira and M. Caus，用于时间序列分析的 *Kolmogorov-Arnold* 网络（*KAN*），*arXiv* 预印本 *arXiv:2405.08790*，（*2024*）。
[62] A. Vaswani，《*Attention Is All You Need*》，《神经信息处理系统进展》，（*2017*）。
[63] S. Wang S. Sankaran X. Fan P. Stinis and P. Perdikaris，使用物理信息神经网络模拟三维湍流，*arXiv* 预印本 *arXiv:2507.08972*，（*2025*）。[64] S. Wang S. Sankaran H. Wang and P. Perdikaris，训练物理信息神经网络的专家指南，*arXiv* 预印本 *arXiv:2308.08468*，（*2023*）。
[65] Y. Wu T. Su B. Du S. Hu J. Xiong and D. Pan，*Kolmogorov–Arnold* 网络使物理定律学习变得简单，《物理化学快报》，*15*（*2024*），第 *12393–12400* 页。[66] R. Yu W. Yu and X. Wang，*KAN* 还是多层感知机：一种更公平的比较，*arXiv* 预印本 *arXiv:2407.16674*，（*2024*）。[67] C. Zeng J. Wang H. Shen and Q. Wang，*KAN* 与多层感知机在不规则或含噪函数上的比较，*arXiv* 预印本 *arXiv:2408.07906*，（*2024*）。

补充材料

C 神经切线核与基变换

神经切线核（NTK）是一个强大的工具，可用于理解无限宽极限下神经网络的训练动态 [31]。NTK 描述了训练过程中网络输出相对于其参数的变化。对于参数为 w 的神经网络 $f(\mathbf{x};\mathbf{w})$，NTK 定义为 $\mathrm{W}(\mathbf{x},\mathbf{x}') := \left\langle \frac{\partial f(\mathbf{x};\mathbf{w})}{\partial \mathbf{w}}, \frac{\partial f(\mathbf{x}';\mathbf{w})}{\partial \mathbf{w}} \right\rangle$。对于包含 k 个数据点的批次，我们可以将这些梯度组织成雅可比矩阵 $J \in \mathbb{R}^{k\times N_w}$，其中每一行包含第 i 个数据点对应的 $\frac{\partial f(\mathbf{x}^{(i)};\mathbf{w})}{\partial \mathbf{w}}$。该批次的 NTK 矩阵由格拉姆矩阵 $\mathrm{NTK} = JJ^T \in \mathbb{R}^{k\times k}$ 给出，其中第 (i,j) 个元素恰好是 $\mathrm{W}(\mathbf{x}^{(i)},\mathbf{x}^{(j)})$。在我们将基从样条权重 u 变换为 ReLU 权重 $\mathbf{w} = A^T\mathbf{u}$ 时，雅可比矩阵按如下方式变换：$J_g = J_f A^T$，从而得到样条基对应的 NTK $\mathrm{NTK}_S^{[r]} = J_f A^{[r]} (A^{[r]})^T J_f^T$。

下面的定理介绍了块托普利茨算子理论领域中的一个具体结果，该结果用于证明接下来的定理。

定理1 (块托普利茨算子的最大奇异值 [58])。 设 $T_N(F)$ 是一个 $N\times N$ 块托普利茨矩阵，其生成函数 $F(x):[0,2\pi]\to\mathbb{C}^{m\times m}$ 连续。那么，最大奇异值的上界为

$$\sigma_{max}(T_N(F)) \le \max_{x\in[0,2\pi]} \sigma_{max}(F(x)),$$

对所有 $N\in\mathbb{N}$ 均成立。

现在，我们来探讨 NTK 矩阵在 ReLU 基和样条基下的谱半径。

定理 2。 对于对称半正定的 M，令 $\rho(M) \ge 0$ 表示谱半径。那么，对于均匀节点间距，

$$\rho(NTK_S^{[r]}) \le 4\rho(NTK_R). \tag{38}$$

证明。 定义一个辅助的等价 ReLU^{r-1} 基 $\varphi_i^{[r]}(x) = \mathrm{ReLU}\left(\frac{x-t_i}{h}\right)^{r-1}$. 由于 ReLU 是关于正标量的齐次函数，因此该基再次等价于 ReLU 基。

因此，我们重复上述构造过程，得到一个等价的基变换矩阵 $\widetilde{A}^{[r]}$，其中 $\widetilde{A}^{[r]} = \frac{1}{h^{1-r}} A^{[r]}$. 令 J 表示 f 关于权重 W 的雅可比矩阵，使得

$$\mathrm{NTK}_R = JJ^T, \qquad \mathrm{NTK}_S^{[r]} = J\mathbb{A}\mathbb{A}^T J^T, \tag{39}$$

其中 $\mathbb{A}$ 是按层特定的 $A^{[r]}$ 重新排序后的块对角算子。

注意， JJ^T 和 $J\mathbb{A}\mathbb{A}^T J^T$ 是对称的，且特征值非负；它们的最大特征值分别是 J 和 $J\mathbb{A}$ 的最大奇异值的平方。注意，算子的最大奇异值由其 ℓ^2 范数给出，根据次乘性， $\|J\mathbb{A}\| \le \|J\|\|\mathbb{A}\|$。因此，我们将通过考察基变换矩阵 $\mathbb{A}$ 的最大奇异值来界定 JA 的谱。回顾一下，在正确的排序下， $\mathbb{A}$ 是块对角矩阵，其每一层的对角块为 $\widetilde{A}^{[r]}$ 。块对角矩阵的最大奇异值等于各块最大奇异值中的最大值。因此，对于任意 P， Q 以及固定的 r，我们将考察 $\|\widetilde{A}^{[r]}\|$ 。

根据推论 1，对于均匀节点， $\widetilde{A}^{[r]}$ 是上三角托普利兹矩阵。借助托普利兹矩阵理论，通过考察算子的生成函数，可以渐近地（关于 n）对 $\widetilde{A}^{[r]}$ 的最大奇异值（因此也等价于 $\widetilde{A}^{[r]}(\widetilde{A}^{[r]})^T$的最大特征值）给出紧确界。令 α_ℓ 表示托普利兹矩阵第 i条对角线上的托普利兹系数，其中 α_0 是主对角线， α_{-1}是第一条次对角线，依此类推。那么该托普利兹矩阵对应于一个傅里叶生成函数，$F(x) = \sum_{\ell=-\infty}^{\infty} \alpha_\ell e^{-\mathrm{i}\ell x}$. 在我们的情形中，$\widetilde{A}^{[r]}$ 的标量（即块大小 $N=1$）生成函数为

$$F_r(x) = r\sum_{\ell=0}^{r} \frac{(-1)^\ell}{(\ell)!\,(r-\ell)!}\, e^{-\mathrm{i}\ell x} = r\cdot\frac{(1-e^{\mathrm{i}x})^r}{r!} = n^{r-1}\frac{(1-e^{\mathrm{i}x})^r}{(r-1)!}. \tag{40}$$

因此，根据定理1，对于 $r \in \mathbb{Z}^+$，我们有

$$\|\widetilde{A}^{[r]}\| \leq \max_x |F_r(x)| = \frac{2^r}{(r-1)!} \leq 4 \quad \text{for } r \geq 1, \tag{41}$$

至此证明完毕。 □

D 超参数与特化

本文中的示例在 Nvidia A100 GPU 上运行，采用了多种不同的问题配置和随机种子。

D.1 函数回归

第 5.1 节中的函数回归示例使用了一个数据集，其中包含网格 $[0.0001, 0.9999]^2$ 上的 20000 个随机均匀点。网络输出通过仿射变换归一化到 $[0, 1]$。随机数生成器种子固定为 1234，任何五次运行的集成均使用随机种子 1234 至 1238。优化器为 L-BFGS，学习率为 1.0。

D.2 物理信息神经网络（PINNs）

二维泊松问题对体积项使用基于残差的注意力机制（RBA） [3] 。假设集合 V 中有 $N_V = |V|$ 个点，并初始化权重向量 $\vec{\lambda}$，使得 $\lambda_i = 1$，其中 $i = 1 \dots N$。RBA方案提供的更新规则为

$$\lambda_i = (1-\mu)\lambda_i + \mu \frac{e_i}{\max_j e_j} \text{ where } e_i = \left| \nabla \cdot (\epsilon(\vec{x}) \nabla u_N(\vec{x})) - f(\vec{x}) \right|. \tag{42}$$

权重向量会缩放体积损失项中的每个点，因此也会被纳入梯度。如果 e_i 相对较大，则该误差对损失的贡献权重也会更大。

多层 KAN 训练算法会在训练开始时以及层级转换期间，采用学习率线性渐增的策略。在表格中，我们用 η_0 表示起始学习率，用 η 表示最终目标学习率。从一个学习率过渡到另一个所需的步数（对于单批次物理信息神经网络训练，则为训练轮数）记为 S_{LR}。

第5.2.1节：二维泊松超参数

参数	符号	值
损失缩放	Γ_V	10^{-2}
	Γ_I	10^{-1}
	Γ_B	10^{0}
体积点	$\|V\|$	2401
边界点	$\|B\|$	200
界面点	$\|I\|$	49
MLP层		$[2, 16, 16, 1]$
KAN层		$[2, 5, 1]$
KAN样条阶数		3
KAN粗网格尺寸		4
优化器		Adam
学习率	η	10^{-2}
初始学习率	η_0	10^{-4}
渐进步骤	S_{LR}	10
RBA 阻尼	μ	10^{-4}

一维 Burgers 示例使用的随机数生成器种子从 1234 开始，每次新运行时递增 10。

第 5.2.2 节：一维 Burgers 超参数

参数	符号	值
损失缩放	Γ_V	10^0
	Γ_B	10^0
体积点	$\|V\|$	4096
初始条件点	$\|B\|$	64
优化器		Adam
学习率	η	10^{-3}
周期	S_{LR}	100
指数衰减参数 γ		0.9995

Allen-Cahn示例使用随机数生成器种子1234。该示例使用Adam优化器，并在前10个训练轮次中通过线性学习率调度器将学习率从 10^{-4} 增加到 10^{-2}。

第 5.2.3 节：Allen-Cahn超参数

参数	符号	值
损失缩放	Γ_V	10^{-1}
	Γ_B	10^0
体积点	$\|V\|$	251001
初始条件点 $\|B\|$		501
优化器		Adam
学习率	η	10^{-2}

E Allen-CahnReLU训练

参见图9。

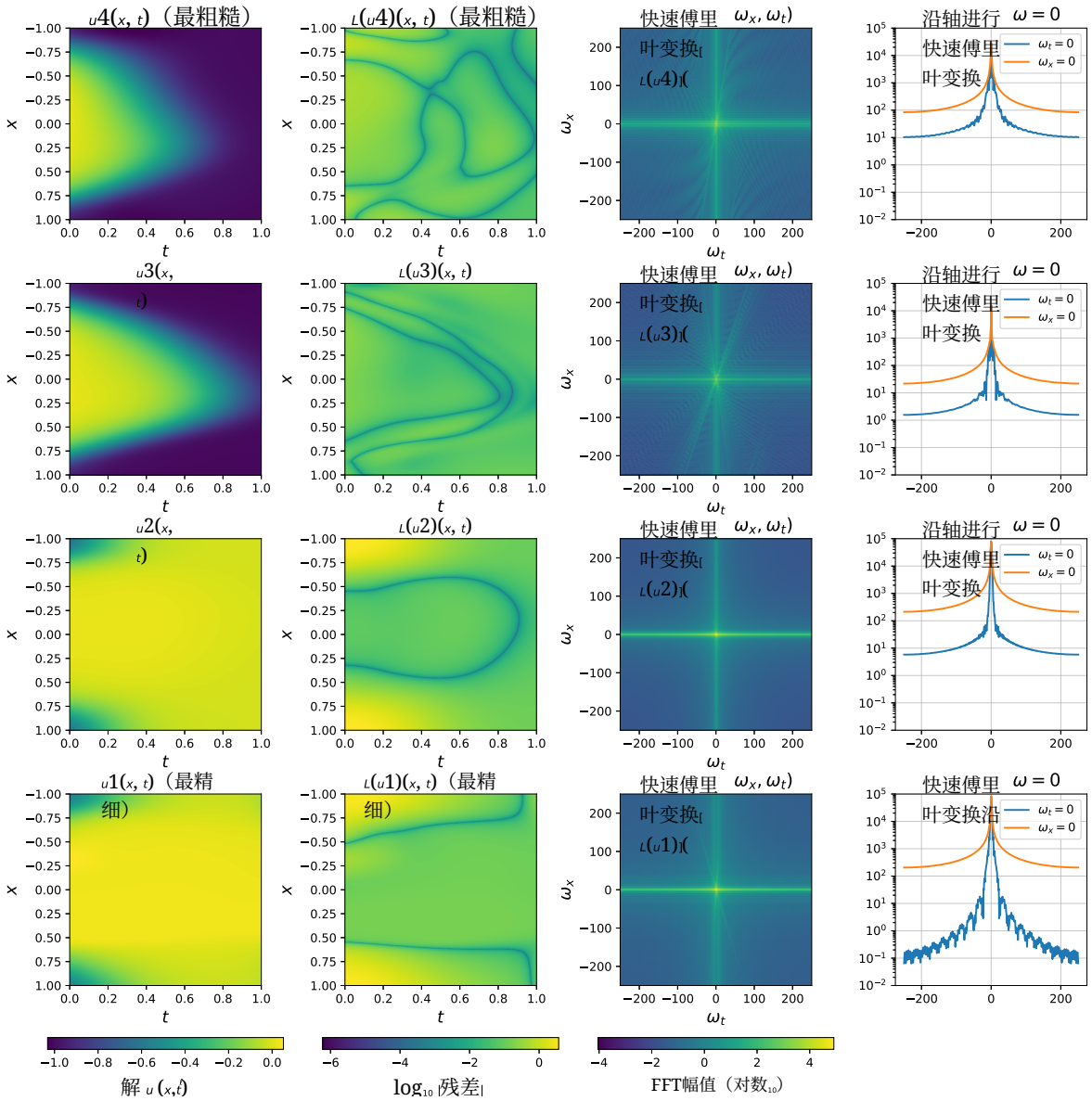


图9：Allen-Cahn方程的训练后 ReLU基解（最左）、对数尺度下的残差（中左）、残差的傅里叶变换（中右），以及沿坐标轴 $\omega_t = 0$ 和 $\omega_x = 0$ 的傅里叶变换截面（最右）。每一行展示了在各个细化层级训练后的输出，顺序从最粗糙（顶部）到最精细（底部）。前三列共用同一色条，显示在各列底部。