arXiv:2302.00807v1 [physics.flu-dyn] 2 Feb 2023

深度神经算子可作为形状优化的精确替代模型：翼型案例研究

***Khemraj Shukla, *Vivek Oommen, *Ahmad Peyvan**布朗大学，普罗维登斯，罗德岛州 02912

***Michael Penwarden**
犹他大学，盐湖城，犹他州 84112

Luis Bravo,AnindyaGhoshal武器与材料局，美国陆军研究实验室，阿伯丁试验场，马里兰州 21005

Robert M. Kirby GeorgeEm Karniadakis犹他大学，盐湖城，犹他州 84112；布朗大学，普罗维登斯，罗德岛州 02912

***同等贡献**

摘要

深度神经算子（如 DeepONets）改变了高维非线性回归的范式，使其从函数回归转向（微分）算子回归，为计算工程应用带来了重大变革。本文探讨了使用 DeepONets 推断未见过的翼型周围流场，以实现形状优化。形状优化是空气动力学中的一项重要设计问题，通常需要耗费大量计算资源。我们给出的结果表明，预测精度几乎没有下降，同时在线优化成本降低了数个数量级。我们将 NACA 翼型作为所提方法的测试案例，因为其形状可以通过四位数参数化轻松定义。我们成功优化了以最大化升阻比为目标的受约束 NACA 四位数问题，并通过与高阶计算流体力学求解器的结果进行比较，验证了所有结果。我们发现 DeepONets 的泛化误差较低，因此非常适合生成未见形状的解。具体而言，只需不到一秒即可准确推断压力、密度和速度场，从而能够使用通用目标函数，而不局限于本文所考虑的最大化升阻比。

***K*关键词** 神经　　算子 · DeepONet · 翼型形状优化 · 纳维-斯托克斯方程 · 替代模型　　ls

术语表

m　　= 弦长百分比表示的最大弯度 p　　= 弦长百分比表示的最大弯度位置 t　　= 翼型最大厚度占弦长的百分比 ρ　　= 无量纲密度 u　　= 流体在x 方向上的无量纲速度 v　　= 流体在y 方向上的无量纲速度

p = 无量纲压力T = 无量纲温度$\mathcal{G}$ = DeepONet 模型的输出θ_b = 分支网络的所有可训练权重和偏置的集合θ_t = 主干网络的所有可训练权重和偏置的集合N_ϕ = DeepONet 学习的基函数数量τ_{ij} = 黏性应力张量的分量f = DeepONet 的广义输入函数ξ_g = 几何参数c = 弦长Ma = 马赫数ξ_f = 流动参数Re = 雷诺数Pr = 普朗特数$\boldsymbol{\alpha}$ = 分支网络的输出$\boldsymbol{\phi}$ = 主干网络的输出τ_w = 壁面剪切应力L =升力D = 阻力$\vec{n}$ = 单位法向量$\vec{t}$ = 单位切向量

1 引言

用于求解回归问题的神经网络将输入数据映射为输出数据，而神经算子则将函数映射为函数。这一视角上的近期范式转变始于关于深度算子网络（DeepONet）的开创性论文 [1, 2], ，它带来了一种全新的建模能力，在工程领域极具实用价值——即能够用神经算子取代复杂且耗费大量计算资源的多物理场系统，并实时提供函数输出。具体而言，与其他在训练和测试期间需要进行优化的物理信息神经网络（PINN） [3] 不同，DeepONet 在推理期间无需任何优化，因此可用于实时预测，包括设计、自主运行、控制等。图 1 展示了 DeepONet 的结构示意图，以及其各组成部分的常用名称。DeepONet 可以在分支网络中接收多保真或多模态输入[4, 5, 6, 7, 8] ，并使用一个独立网络作为主干网络；该网络以连续形式表示输出空间，例如时空坐标或参数空间。从某种意义上说，DeepONet 可以像降阶模型（ROM） [9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19]一样用作替代模型。然而，与 ROM 不同，DeepONet 采用过参数化设计，因此具备 ROM 所不具备的泛化能力和抗噪鲁棒性，参见近期研究 [20]。

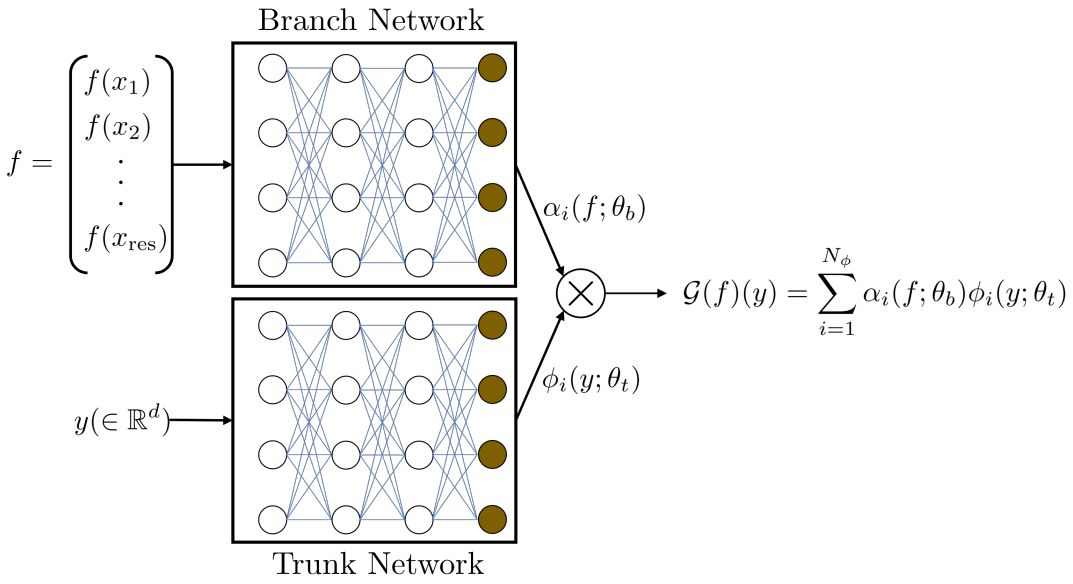


图 1：**DeepONet 示意图**。 f 表示提供给 DeepONet 的一般输入函数。将分辨率为（=res）的 f 离散表示作为分支网络的输入。在此图中， y（$\in \mathbb{R}^d$）表示输出函数的区域坐标。

在本研究中，我们探讨了使用 DeepONet 表示不同解域上的函数的可能性；此前尚未对此进行研究。具体而言，我们聚焦于气动设计，研究亚声速流动条件下翼型形状优化这一经典问题。翼型气动外形优化（ASO）在高效现代商用飞机的设计中发挥着至关重要的作用。气动几何优化过程通常针对构成三维（3D）翼型横截面的翼型形状进行。几何优化旨在降低阻力、提高升力，从而提升飞机的燃油效率，进而降低运输成本。优化过程通常需要一个数值模型，用于预测给定翼型几何形状周围的流场，并计算指定流动条件下的升力和阻力。传统上，这类数值模型采用可压缩流数值求解器；要准确模拟复杂翼型周围的流场，此类求解器的计算开销很大。可以引入代理模型，绕过优化循环中由数值求解器计算气动力这一耗时环节。

气动外形优化传统上采用两种主要范式：基于梯度的优化和无梯度优化方法。基于梯度的方法需要计算代价函数相对于设计变量的导数。当设计变量的数量超过某一阈值时，由于计算成本高昂，基于梯度的优化将变得不可行 [21]。伴随公式由欧拉方程或纳维-斯托克斯方程推导而来，使梯度优化不受设计变量数量的影响 [22, 23, 24, 25]。求解伴随方程可能与求解控制方程一样耗时（也就是说，经验法则是，正向求解与伴随求解的总成本至少是正向求解器成本的两倍）。此外，优化过程可能陷入

局部极小值，导致几何形状未得到优化 [26]。无梯度方法 [27, 28]可以通过采用直接优化方法来避免局部极小值；该方法需要对可压缩流进行大量且代价高昂的数值模拟。大量流动仿真有助于找到代价函数的全局最小值，但代价是计算成本高昂。代理模型能够以可接受的精度再现流场，与完整的 CFD 仿真相比，速度可大幅提升。随后，代理模型可用于基于梯度或无梯度的全局优化过程。基于代理模型的方法通常与无梯度优化方法结合使用，例如遗传算法和粒子群优化（PSO） [29] 方法。Krige [30] 提出了克里金代理模型，该模型应用于航空航天设计实验 [31, 27]。克里金代理模型在优化过程开始前和进行中都必须接受训练，因为仅依靠初始训练得到的代理模型通常并不准确。克里金代理模型的泛化误差较高，导致模型在初始训练阶段的预测不准确。本文提出的深度算子网络（DeepONet）能够缓解这一问题，因为它将一个函数映射到另一个函数，从而显著降低泛化误差 [1]。

几何形状的参数化会极大地影响气动外形优化的计算成本和精度。几何形状参数化能够减少优化算法需要搜索的设计变量数量。减少设计变量数量可以简化优化过程，并降低对噪声的敏感性。参数化模型必须能够再现多种翼型形状，同时将设计变量数量保持在最低限度。气动外形优化中已采用多种几何参数化模型。Carpentieri等人 [23] 采用正交切比雪夫多项式构造翼型曲线。采用正交多项式覆盖整个设计空间。Lepine等人 [32] 仅使用13个控制点，便通过非均匀有理B样条对一大类翼型形状进行参数化。遵循Lepine等人 [32]的思路，Srinath和Mittal [25]也采用非均匀有理B样条进行参数化。其他研究人员则使用B样条 [33], 和贝塞尔 [34] 曲线，以及Hicks和Henn函数 [35]来构造翼型形状。Painchaud-Ouellet等人 [36]采用非均匀有理B样条，在跨声速范围内对翼型进行形状优化。他们指出，使用非均匀有理B样条可以确保翼型形状的正则性。翼型形状也可以通过变形方法 [35]构造。该方法将多个凸起线性组合后叠加到基准翼型形状上，以实现参数化 [37, 38]。类别函数/形状函数变换方法 [39] 采用伯恩斯坦多项式 [40] 对翼型及其他空气动力学几何形状进行参数化。另有研究采用本征正交分解（POD） [39] 来减少设计变量的数量。本研究采用NACA四位数翼型和非均匀有理B样条参数化表示来构造翼型形状。

随着计算能力的显著提升，深度神经网络（DNN）工具在代理模型开发中受到广泛关注 [41, 42, 43]。深度神经网络方法可针对大量输入设计变量进行训练，从而预测优化循环中的代价函数。Du 等人 [44] 训练了一个前馈深度神经网络，以接收翼型形状并预测阻力系数和升力系数。他们还使用循环神经网络模型估算压力系数。利用代理模型确定的最优翼型设计，与通过基于 CFD 的优化过程获得的翼型设计进行了比较 [44]。Liao等人 [45] 利用多保真卷积神经网络（CNN）和迁移学习设计了一个代理模型。这种学习方法将特定领域中学到的信息迁移到相似领域。低保真样本作为源数据，高保真样本则作为目标数据。Tao 和 Sun [46] 引入了一个使用低保真数据训练的深度置信网络（DBN）。随后，他们通过回归将训练后的 DBN 与高保真数据结合，建立了用于形状优化的代理模型。现有的形状优化代理模型均经过训练，用于预测升力、阻力或压力系数；相比之下，它们并不推断气动外形周围的流场。在本研究中，我们利用 DeepONet 构建代理模型，以预测翼型形状周围的流场。预测流场能够提供额外信息，可用于优化循环中的代价函数。我们的目标是开发一种气动外形优化框架，利用代理模型推断几何形状周围的流场。该代理模型基于 DeepONet 构建，并使用亚音速流动状态下翼型的高保真 CFD 仿真进行训练。随后，我们将该代理模型应用于两个不同的形状优化框架。本研究的创新点包括：

- 构建基于 DeepONet 的代理模型，作为昂贵 CFD 求解器的一种高效且低成本的实现。
- 代理模型的构建不受输入空间的影响，输入空间可以定义为低维或高维参数化表示。
- 高维流场的预测可用于优化循环中的各种代价函数。
- 利用推断出的高维流场计算阻力系数和升力系数，从而获得更准确的预测。
- 将 Dakota 优化框架与 DeepONet 代理模型相结合。

本文其余部分安排如下。首先，我们定义优化过程。接着，我们介绍如何使用开源谱元 Nektar++ 计算流体力学求解器生成用于训练代理模型的数据。然后，我们说明基于 DeepONet 的代理模型的训练流程。随后，我们展示采用两种不同方法得到的优化结果。最后，我们在结论部分总结研究发现。在附录中，我们通过使用不同代码重复选定的仿真，验证所生成数据的准确性。此外，我们通过与多种方法进行比较来验证 Dakota 优化器，并发现所研究的所有方法都收敛到相同的解。

2 问题设置

为了凸显 DeepONet 作为函数到函数映射在翼型形状优化过程中发挥作用的能力，我们首先回顾结合 DeepONet训练的传统端到端形状优化流程。图 2 展示了该流程的示意图。从左上方到右下方依次来看，我们首先设置实验条件。这包括确定训练中参数化翼型的可行集、空气动力学条件，以及与问题相关的其他工程约束。我们选择 NACA 四位数翼型作为几何表示，它根据随机抽取的参数给出上、下表面方程。该表示有三种用途：（1）直接对翼型周围的流场划分网格，此处使用 Gmsh；（2）在预测目标升阻比时，用于在翼型表面查询代理模型；（3）作为 DeepONet 函数到函数映射的分支输入，并使用非均匀有理B样条进行预处理以降低维数。

在创建代理模型方面，这可以视为以下正向问题：首先确定几何参数和流动参数，然后将它们用于构造计算流体力学求解器的输入：翼型的几何表示及其对应的网格，用于近似流场；以及一个流动参数文件。随后将这些输入流动求解器——在我们的案例中，使用的是计算流体力学求解器 Nektar++。该求解器的结果用于生成训练数据，以训练我们的 DeepONet 代理模型。最后，图的右下象限表示使用训练好的 DeepONet 代理模型进行形状优化的过程。该过程是递归的：优化器（我们使用 Dakota）会不断向 DeepONet 代理模型查询新的设计参数，直到目标函数得到充分最小化。与现有方法不同的是，这一优化过程的大部分工作都在线下完成。使用计算流体力学求解器生成数据可能成本高昂，但借助最终训练好的 DeepONet，几何优化的在线成本比其他方法快几个数量级。此外，只要目标能够根据 DeepONet 所训练的流场构造，就可以定义不同于升阻比的目标函数，并针对新目标快速优化翼型，无需增加任何额外成本。

在我们的实验中，我们重点使用雷诺数为 $Re = 500$ 、马赫数为 $Ma = 0.5$ 的二维可压缩纳维–斯托克斯场进行训练。选择这些数值是为了重点考察 DeepONet 捕捉区域变化的能力（而非非定常性等因素带来的叠加影响）。鉴于 DeepONets 在这一实验设置下表现出色，未来的工作将把这一流程扩展到更复杂的流动和实验条件，例如改变攻角、采用变形几何、处理非定常流动，或拓展到高速流动状态。

3 方法

3.1 数据生成

对于数据集中的每个样本，我们定义一组几何参数（ξ_g）和流动参数（ξ_f）。随后将几何参数转换为另一种表示形式，例如根据 NACA 翼型方程推导出的表面坐标；该变换由 $\Gamma(\xi_g)$ 给出。接着使用这些点，通过 Gmsh[47]，对流场区域生成网格，再将其输入流场模拟软件 Nektar++ [48, 49]。通过后处理，我们从 Nektar++ 获得密度、x 方向速度、y 方向速度和压力的场解。结果分为两组保存：一组位于翼型周围的子域中，用于训练 DeepONet；另一组位于翼型表面的传感器处，用于验证。我们还保存 Nektar++ 得到的升力和阻力，以验证翼型力的离散积分。

3.1.1 几何生成

NACA 四位数翼型是使用 DeepONets 进行几何优化的理想测试案例，因为它们可以通过广为人知且经过充分研究的参数化几何方程来表示多种形状。我们的几何优化框架可以轻松扩展到任何参数化几何，例如 NACA 五位数翼型及其他类型。

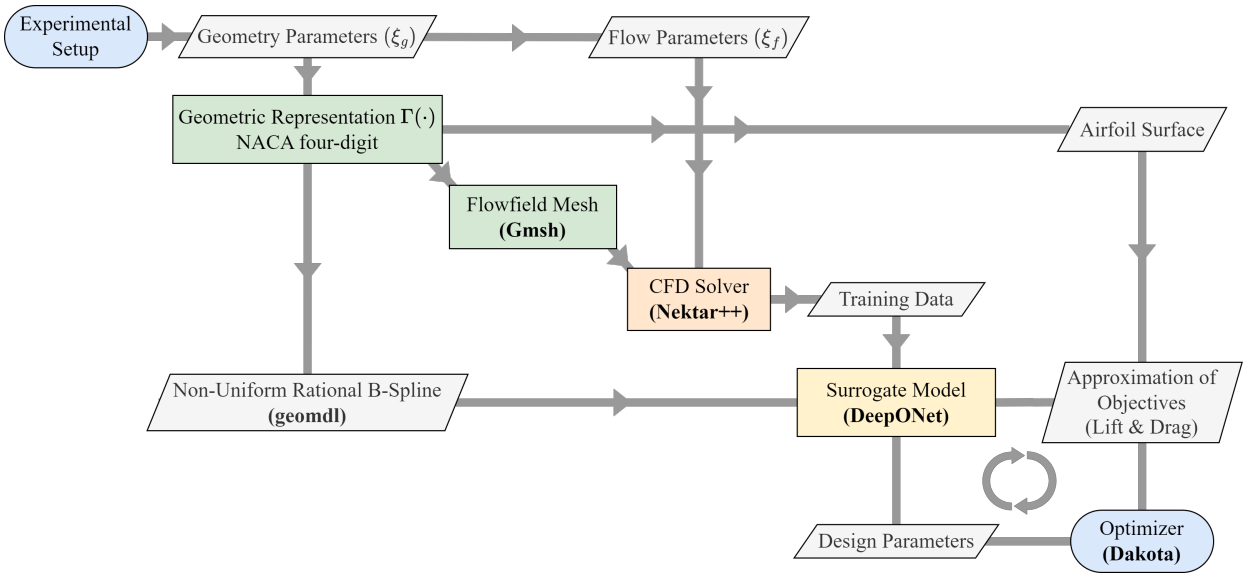


图 2：使用 DeepONet 代理模型进行翼型几何优化的示意图。蓝色椭圆表示该方法的初始状态和最终状态。主要中间步骤以彩色方框突出显示，辅助步骤则以灰色平行四边形表示。

按照 [50],，我们将弦长为 1 的翼型表面的参数方程定义如下：

$$y_t = \frac{t}{0.2}\left(a_0\sqrt{x} + a_1 x + a_2 x^2 + a_3 x^3 + a_x x^4\right) \tag{1}$$

$$y_c = \begin{cases} \frac{m}{p^2}\left(2px - x^2\right) & \text{如果 } x < p \\ \frac{m}{2}\left(1-2p+2px-x^2\right) & \text{如果 } x > p \end{cases} \qquad \theta = \tan^{-1}\left(\frac{dy_c}{dx}\right) \tag{2}$$

$$x_u = x - y_t \text{正弦}(\theta), \quad y_u = y_c + y_t \text{余弦}(\theta)$$
$$x_l = x + y_t \sin(\theta), \quad y_l = y_c - y_t \cos(\theta) \tag{3}$$

其中 $a_0 = 0.2969$，$a_1 = -0.1260$，$a_2 = -0.3516$，$a_3 = 0.2843$，$a_4 = -0.1015$。因此，我们可以将几何形状定义为一个点云，其坐标集为（x_u，y_u，x_l，y_l），由 $\xi_g = (t,\ p,\ m)$ 参数化。使用余弦间距在 x 个位置上取点，共 100 个点；这提高了前缘和后缘附近的几何保真度，从而提升了这些关键位置处网格和流场模拟的准确性。为简化问题并降低发生流动分离或湍流的可能性，我们对 ξ_g 施加约束。最大厚度（t）设为常数 0.15，而由最大弯度位置（p）和最大弯度（m）决定的参数空间区域为 $p \times m \in [0.2, 0.5] \times [0.0, 0.09]$。因此，对于训练集或测试集中的一个几何形状样本，我们抽取一个 ξ_g 元组，其中各变量参数均从其所属区域内的均匀分布中抽取。我们重复抽取 50 次，根据方程 1–3 得到表面坐标，并将其分为 40 个训练样本和 10 个测试样本，如图 3 所示。训练集与测试集的划分是深度学习的一个关键环节；在这里，我们选择相对稀疏的采样，以突出 DeepONets 对未见参数的良好泛化能力。

接下来，为了降低输入到 DeepONet 分支的维数，我们使用 geomdl [51]，通过 30 个控制点构成的非均匀有理B样条（NURBS）拟合翼型表面。这将输入维数从 200 对（x，y）减少到仅 30。

3.1.2 网格生成

使用 Gmsh [47] 为参数化翼型几何生成网格，在参数化位置处的最小特征长度为 0.01。翼型流场的网格采用直接根据 NACA方程生成的 200 个表面点，而非存在一定误差的非均匀有理B样条拟合。使用样条函数表示翼型边界的一维几何形状。为解析前缘和后缘处的流动，采用分裂方法 [52]对网格进行加密。非均匀有理B样条低维表示是减少 DeepONet 过度参数化的一种手段，但生成高保真训练数据并不需要这一步。利用集成了 Python 几何生成流程的 Gmsh Python API，自动完成全部 50 个翼型的网格生成，无需人工操作。图

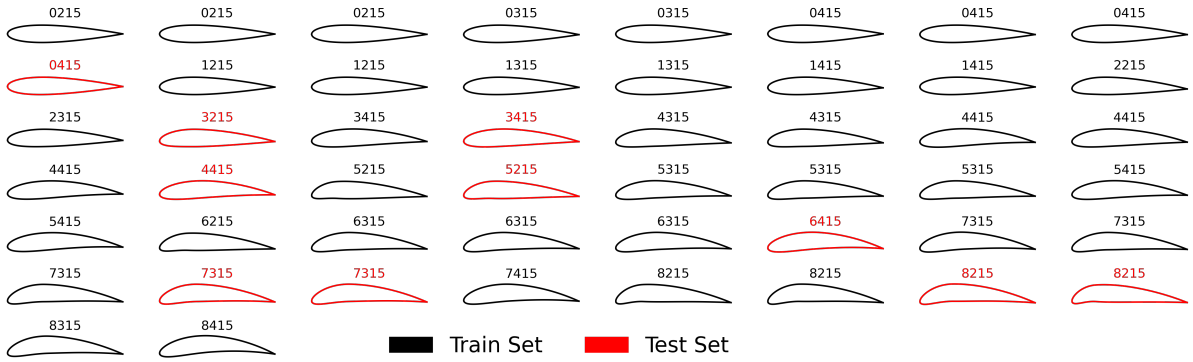


图3：在 ξ_g 域中采样的训练集和测试集几何形状。请注意，编号对应 NACA翼型；编号重复是由于四舍五入到最接近的整数以便于阅读。真实的 NACA参数取自均匀分布，因此为实数。

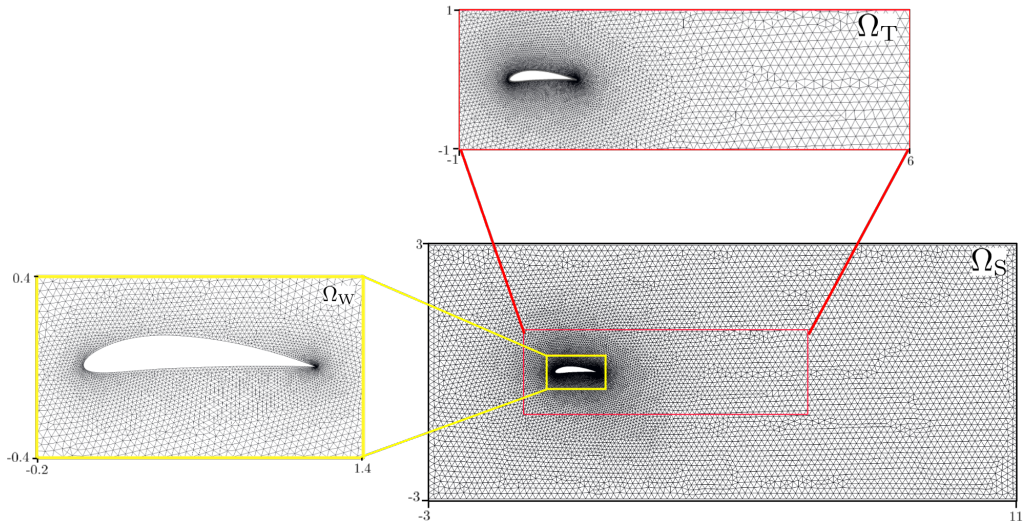


图4：带包围域的翼型离散化。Ω_S 表示求解可压缩纳维-斯托克斯方程的整个域。Ω_T （红色边框的插图）表示用于训练 DeepONet 的域。黄色边界的插图表示子域（Ω_W），展示了翼型周围的网格加密。

4 展示了整个模拟域 Ω_S的网格，随后将其与流动参数一起输入 Nektar++ ，以生成 DeepONet训练数据。如图所示，求解得到的稳态域 Ω_T中只有一部分用于训练 DeepONet。这是因为 DeepONet 是一种函数到函数的映射，不会严格遵循边界条件，也不受边界反射等现象的影响。它对数据集进行回归，而非求解方程组，因此可以采用较小的计算域，甚至不必包含自由来流条件。由于目标是几何优化，这个子域简化了 DeepONet 的训练问题，并降低了训练成本。

3.1.3 流场模拟

我们使用在 Nektar++ 中实现的可压缩流动求解器生成流场数据。Nektar++(www.nektar.info) 是一个开源软件框架， [48, 49] 旨在支持使用谱/hp 元方法开发高性能、可扩展的偏微分方程求解器。二维可压缩流

i动求解器使用二维可压缩 Navier–Stokes 方程，其表达式为：

$$
\begin{aligned}
&\frac{\partial \rho}{\partial t}+\frac{\partial \rho u}{\partial x}+\frac{\partial \rho v}{\partial y}=0\\
&\frac{\partial \rho u}{\partial t}+\frac{\partial \rho u^2+p}{\partial x}+\frac{\partial \rho uv}{\partial y}=\frac{1}{Re}\left(\frac{\partial \tau_{xx}}{\partial x}+\frac{\partial \tau_{yx}}{\partial y}\right)\\
&\frac{\partial \rho v}{\partial t}+\frac{\partial \rho uv}{\partial x}+\frac{\partial \rho v^2+p}{\partial y}=\frac{1}{Re}\left(\frac{\partial \tau_{xy}}{\partial x}+\frac{\partial \tau_{yy}}{\partial y}\right)\\
&\frac{\partial E}{\partial t}+\frac{\partial (E+p)u}{\partial x}+\frac{\partial (E+p)v}{\partial y}=\frac{1}{Re}\left[\frac{\partial \left(u\tau_{xx}+v\tau_{xy}+\kappa\frac{\partial T}{\partial x}\right)}{\partial x}+\frac{\partial \left(u\tau_{xy}+v\tau_{yy}+\kappa\frac{\partial T}{\partial y}\right)}{\partial y}\right],
\end{aligned}
\tag{4}
$$

其中 $p=R\rho T$、 $R=\frac{1}{\gamma Ma^2}$、 $k=\frac{\gamma}{\gamma-1}\frac{\mu R}{Pr}$， μ为无量纲黏度，并根据 Sutherland 定律计算如下

$$
\mu=\frac{T^{3/2}}{Re}\frac{1+C/T_\infty}{T+C/T_\infty}. \tag{5}
$$

τ_{xx}、 τ_{yy}和τ_{xy} 的表达式如下

$$
\begin{aligned}
\tau_{xx}&=2\mu\left(u_x-\frac{u_x+v_y}{3}\right),\\
\tau_{yy}&=2\mu\left(v_y-\frac{u_x+v_y}{3}\right),\\
\tau_{xy}&=\mu\left(u_y+v_x\right).
\end{aligned}
$$

我们旨在通过求解方程 (4) 给出的可压缩纳维–斯托克斯方程，模拟流体绕翼型的流动；自由来流参数为 $M_\infty=0.5$、 $Re_{L=1}=500$， $u_\infty=1$， $v_\infty=0$， $T_\infty=1$，迎角为 $=0$，且 $Pr=0.72$。流动域为 $[-3,11]\times[-3,3]$，并采用协调三角形单元进行离散。为求解方程 (4)，我们使用不连续 Galerkin 谱元方法（DGSEM），其二维基函数由二阶 Legendre 多项式张成。对于对流项和扩散项，空间离散采用弱形式和基于内部惩罚的不连续 Galerkin 方法，并使用 Roe 上风格式。对流项和扩散项采用对角隐式 Runge–Kutta（DIRK）方法进行时间积分。流入边界、流出边界、翼型表面的绝热壁边界条件，以及顶部和底部的高阶边界条件均以弱形式施加。有关求解器和方法的详细介绍，建议读者参阅 [53] 的文章。

3.2 代理模型：DeepONet

3.2.1 DeepONets 简介

神经算子是基于通用算子逼近定理[54]开发的神经网络模型。神经算子能够学习函数空间之间的映射，并直接利用现有训练数据学习底层算子。DeepONets [1] 和傅里叶神经算子（FNO） [55] 是两种广受欢迎的神经算子，广泛用于解决不同科学领域中的各类问题。DeepONet由一个对输入函数进行编码的分支网络和一个学习一组基函数的主干网络组成。DeepONet的输出通过计算分支网络与主干网络输出的内积得到。

3.2.2 DeepONets 的训练与测试

我们训练了四种不同的DeepONet模型，利用训练数据学习给定翼型几何形状（ξ_g）对应的压力（p）、密度（ρ）和速度（u、 v）场。主干网络学习一组基函数（$\boldsymbol{\phi}$）关于空间坐标的函数，而分支网络学习相应系数（$\boldsymbol{\alpha}$）关于翼型几何形状的函数。DeepONet的输出定义为

$$
\mathcal{G}^q(\xi_g)(x,y)=\sum_{i=1}^{N_\phi}\alpha_i(\xi_g;\theta_b^q)\phi_i(x,y;\theta_t^q)\quad q\in\{p,\rho,u,v\}. \tag{6}
$$

对于翼型，可以直接将几何参数ξ_g输入分支网络，从而提供其几何形状。然而，对于一般的任意几何形状， ξ_g 不一定能够显式存在。在这种情况下，几何形状

表2：NURBS-DeepONet 和参数 DeepONet 的超参数

NURBS-DeepONet	
分支网络架构:	[30,100,100,50]
分支网络激活函数:	tanh
主干网络架构:	[2,100,100,50]
主干网络激活函数:	tanh
N_ϕ:	50
优化器:	Adam
学习率:	1.00E-04

Parameter-DeepONet	
分支网络架构:	[2,100,100,50]
分支网络激活函数:	tanh
主干网络架构:	[2,100,100,50]
主干网络激活函数:	tanh
N_ϕ:	50
优化器:	Adam
学习率:	1.00E-04

通常使用NURBS控制点表示。为展示在这两种情况下使用DeepONet的有效性，我们研究了parameter-DeepONet（直接将 ξ_g 作为分支网络输入）和NURBS-DeepONet（将翼型几何形状的NURBS控制点作为分支网络输入）。本研究所用DeepONet的超参数见表2。

3.2.3 升力与阻力计算

我们提出了两种评估目标函数的方法——在本例中，目标函数为升阻比。训练好的 DeepONets 所构建的映射以输入几何形状和输出空间中的空间位置（x， y）为输入，并返回该位置处的场值。因此，要估算升力和阻力，只需对翼型表面进行离散积分；根据方程 1–3，可轻松生成（x， y）点，用于查询目标函数。方程 8 和 9 给出了升力和阻力的离散积分，其中壁面剪切应力按方程 7 近似。

$$\tau_w = \mu \frac{dU}{d\overrightarrow{n}} \tag{7}$$

$$L = \int dF_x = \sum p\overrightarrow{n_x} dS + \sum \tau_w \overrightarrow{t_x} dS \tag{8}$$

$$D = \int dF_y = \sum p\overrightarrow{n_y} dS + \sum \tau_w \overrightarrow{t_y} dS \tag{9}$$

对于气动力中的第一项，压力直接取自 DeepONet 的某个预测结果。对于第二项，黏度 μ(p， ρ) 根据方程 5，由压力和密度 DeepONets 计算得出。最后，翼型表面上流动速度的变化 $\frac{dU}{d\overrightarrow{n}}$ 可通过以下两种方式获得：

1. 有限差分**(A)**：

$$\frac{dU}{d\overrightarrow{n}} = \frac{-U_2 + 4U_1 - 3U_0}{2h} \tag{10}$$

2. 自动微分**(B)**：

$$\frac{dU}{d\overrightarrow{n}} = (v_y - u_x) sin\theta cos\theta - v_x sin^2\theta + u_y cos^2\theta \tag{11}$$

其中， $h = 0.001$ 和 θ 是 x-y 轴与各线段的法向-切向轴之间的夹角。方法 (A) 是一种二阶前向有限差分近似，通过在由表面法线和间距 h确定的适当位置，对 x 方向速度和 y 方向速度 DeepONets 进行采样而得到。方法 (B) 利用了如今广为人知的自动微分技术 [56], ；该技术主要用于物理信息机器学习，以近似偏导数并求得偏微分方程残差。在这里，由于 DeepONet 直接接收空间坐标（x， y）并输出速度分量（u， v），计算图是完整的，因此可以通过此方法估算偏导数（u_x， u_y， v_x， v_y）。图 5 展示了每种方法所需的采样。虽然方法 (A) 的点评估次数是方法 (B) 的三倍，但方法 (B) 需要计算梯度，因此两者的计算成本可视为相近。不过，若映射及后续计算图设置得当，以这种方式使用自动微分的灵活性或许能在未来支持更复杂的目标函数。

4 结果

4.1 DeepONet模型的结果

表3报告了NURBS和基于参数的DeepONet在四种不同场上的训练和测试相对 L^2 误差。我们观察到，NURBS和基于参数的DeepONet对各场的预测精度相近

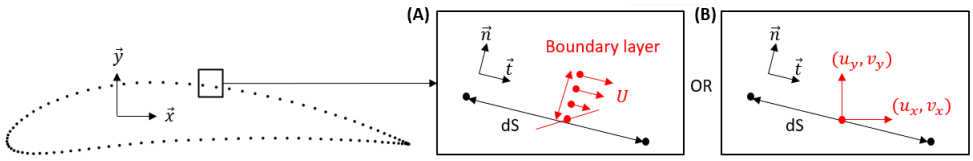


图 5：升力和阻力离散积分示意图。红色点表示用于构造 $\frac{dU}{d\vec{n}}$ 近似的代理模型样本，其中梯度采用有限差分近似，或采用基于直接网络梯度的自动微分近似。

表 3：训练所得 DeepONet 模型的状态变量相对 L^2 误差

	NURBS-DeepONet		Parameter-DeepONet	
	训练相对 L^2 误差	测试相对 L^2 误差	训练相对 L^2 误差	测试相对 L^2 误差
$\mathcal{G}^p$	4.68e-03	6.05e-03	5.23e-03	6.85e-03
$\mathcal{G}^u$	4.97e-03	6.21e-03	4.12e-03	5.38e-03
$\mathcal{G}^v$	3.73e-03	4.60e-03	3.31e-03	4.25e-03
$\mathcal{G}^\rho$	4.57e-03	5.89e-03	4.00e-03	5.18e-03

。NURBS-DeepONet对压力场的预测略好，而parameter-DeepONet对速度场和密度场的预测略好。主要结论是，对于本实验中的几何优化，这两种表示方式都足够。不过，我们还必须考虑到，未来可能会使用更复杂的几何形状，尤其是涉及局部变形的情况。因此，NURBS表示很可能是必要的，因为直接进行参数映射可能会遗漏局部细节。图7展示了表现最佳的DeepONet模型对流场参数的预测场及其绝对逐点误差。可以看出，总体预测通常较为准确；误差主要集中在翼型前缘。未来将采用自适应加权方案来改进DeepONet训练，尤其针对表面和前缘等难以预测的位置。图6展示了整个数据集上各场对应的相对 L^2 误差。我们观察到泛化误差极小，且DeepONets整体预测准确。

对于几何优化，我们不仅要关注全局流场精度，还要特别关注翼型表面的精度。图 8 展示了图 7 中同一翼型的相应表面预测图，横坐标为沿翼型的 x 方向。我们观察到压力场和密度场的预测精度很高，因此能够非常准确地预测升力和阻力分量

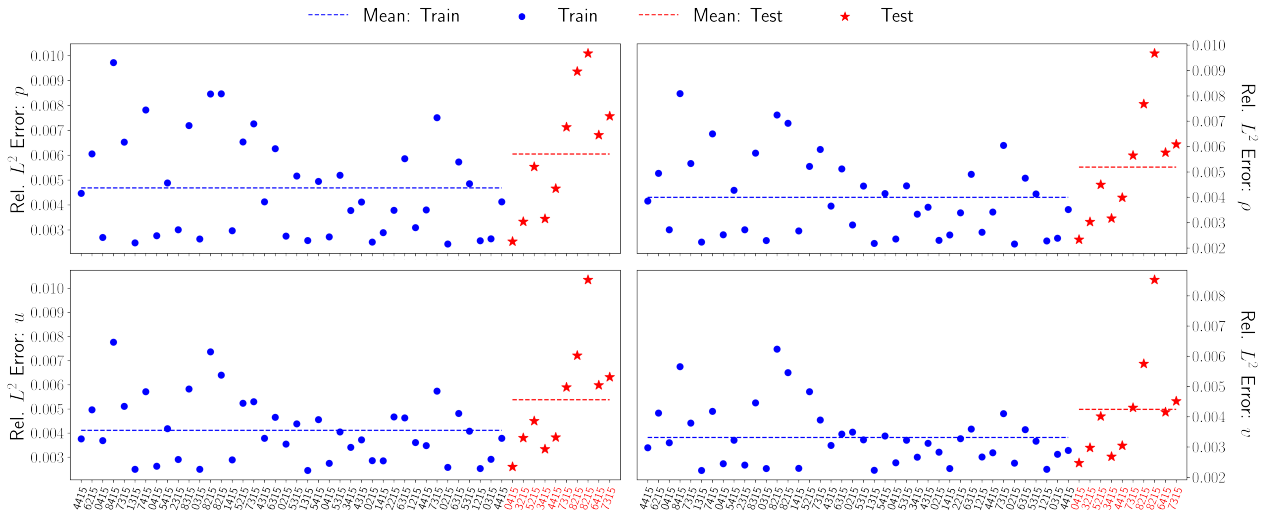


图 6：误差散点图。该图展示了压力场、速度场和密度场对应的每个训练样本和测试样本相对于 DeepONet 预测结果的相对 L^2 误差。

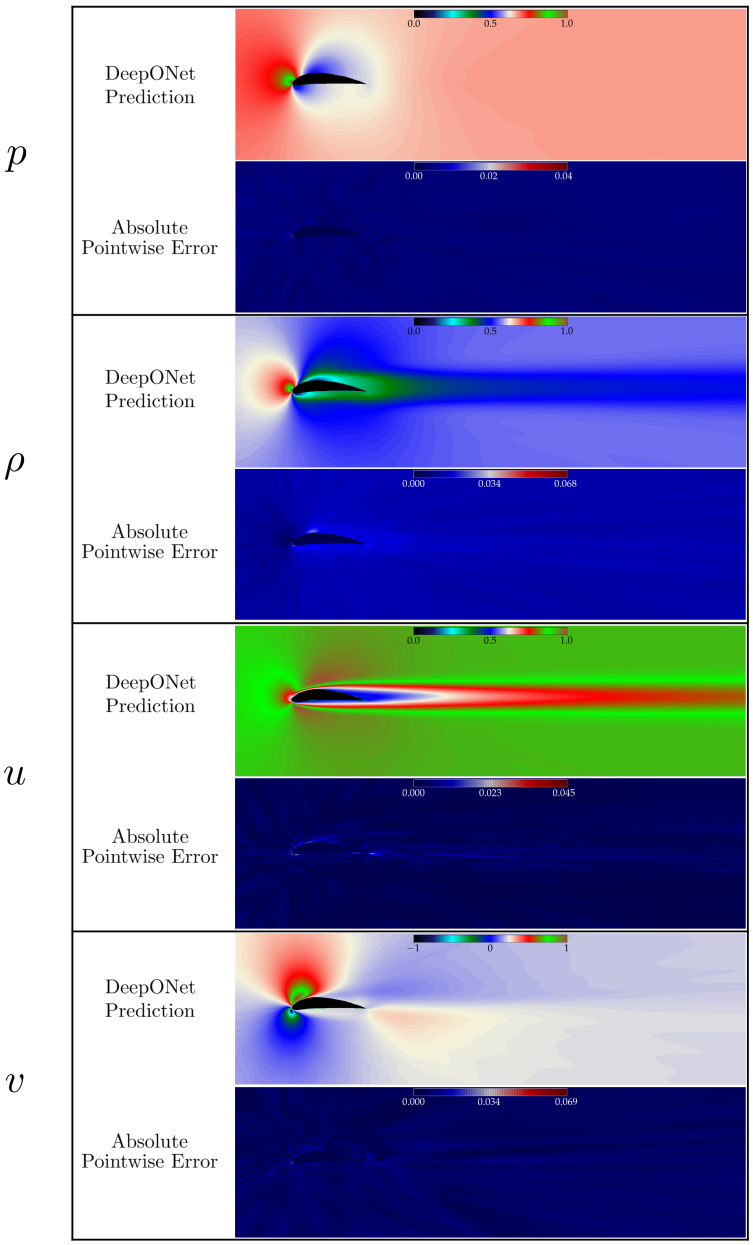


图 7：DeepONet 预测结果。DeepONet 预测的测试集翼型 NACA 7315 周围的压力、密度和速度场，以及相应的逐点绝对误差。

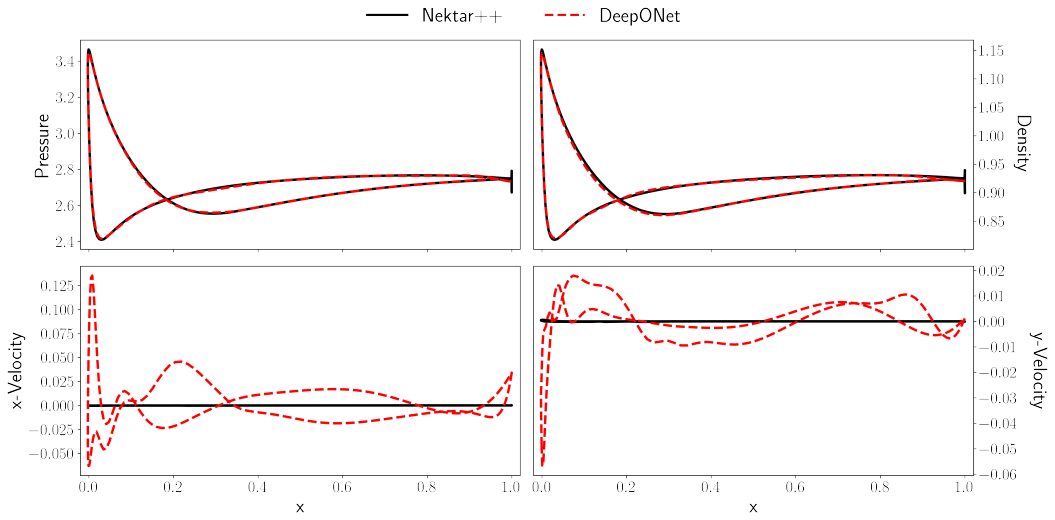


图 8：测试集翼型 NACA **7315**表面上的流场变量图。所有图均显示了表面上的准确预测结果，随后利用这些结果计算升力和阻力。DeepONet 未直接强制施加无滑移条件，因此速度图存在差异。然而，从 y 轴刻度可以看出，除前缘外，预测值接近于零，并不会显著影响总体升力和阻力的计算。

这是由于压力以及黏度 $\mu(p,\ \rho)$所致，而根据公式 5，黏度是这些场的函数。表面的 x和 y 速度场并不一致，因为 DeepONet 没有严格遵守无滑移条件。然而，除前缘外，预测值都接近于零。此外，这些场与目标升阻比并无直接关系，而是通过对表面流速变化的估计 $\frac{dU}{d\vec{n}}$ 间接相关，该估计由公式 10 和 11 中的方法（A）和（B）获得。因此，这种不准确性不会显著影响总体目标预测结果。图 9 进一步证实了这一点，该图展示了整个数据集按升阻比排序后的结果。两种数值积分方法（A）和（B）的结果与数据生成步骤中存储的 Nektar++ 升阻比结果都非常准确。我们还可以从误差图中看到，误差相当均匀，在几何参数空间中没有明显偏差，这表明我们已经充分学习了整个空间，因此最终优化结果是准确的。

4.2 形状优化结果

形状优化的目标是在参数可行域内最大化升阻比；本例中的参数为 m 和 p。方程 12 将这一目标表示为最小化问题，这也是大多数执行基于梯度或无梯度优化的优化器所采用的标准形式。因此，翼型的约束优化问题定义为

$$\begin{aligned} \underset{m,p}{\text{minimize}} \quad & -f(m,p) \\ \text{subject to} \quad & m_{\min} \le m \le m_{\max}, \\ & p_{\min} \le p \le p_{\max}, \end{aligned} \tag{12}$$

其中，$f(m,\ p)$ 表示升力与阻力之比，$[m_{min}, m_{max}]$ 和 $[p_{min}, m_{max}]$ 为可行搜索区域的边界。

本研究的目标之一是针对任意几何形状优化其外形。因此，我们将基于 DeepONet 的代理模型与 Dakota 集成。Dakota 是一个多层级并行面向对象框架，用于设计优化、参数估计、不确定性量化和敏感性分析 [57]。Dakota 可免费使用，并提供高效且可扩展的实现。如算法 1 所示，我们采用模块化方法将 DeepONet 与 Dakota 集成，其中 $\mathcal{D}$ 是从 Dakota 提供的一组优化器中选取的算法，而基于 DeepONet 的模型 Φ 则作为参数传递给 $\mathcal{D}$。例如，为了求解方程（12），我们采用高效全局算法（EGO）。这是一种无导数方法，利用高斯过程模型优化期望改进函数，并基于 NCSU Direct 算法 [58]。选择这种方法的原因是避免调整各种超参数。为使用该算法求解问题，我们

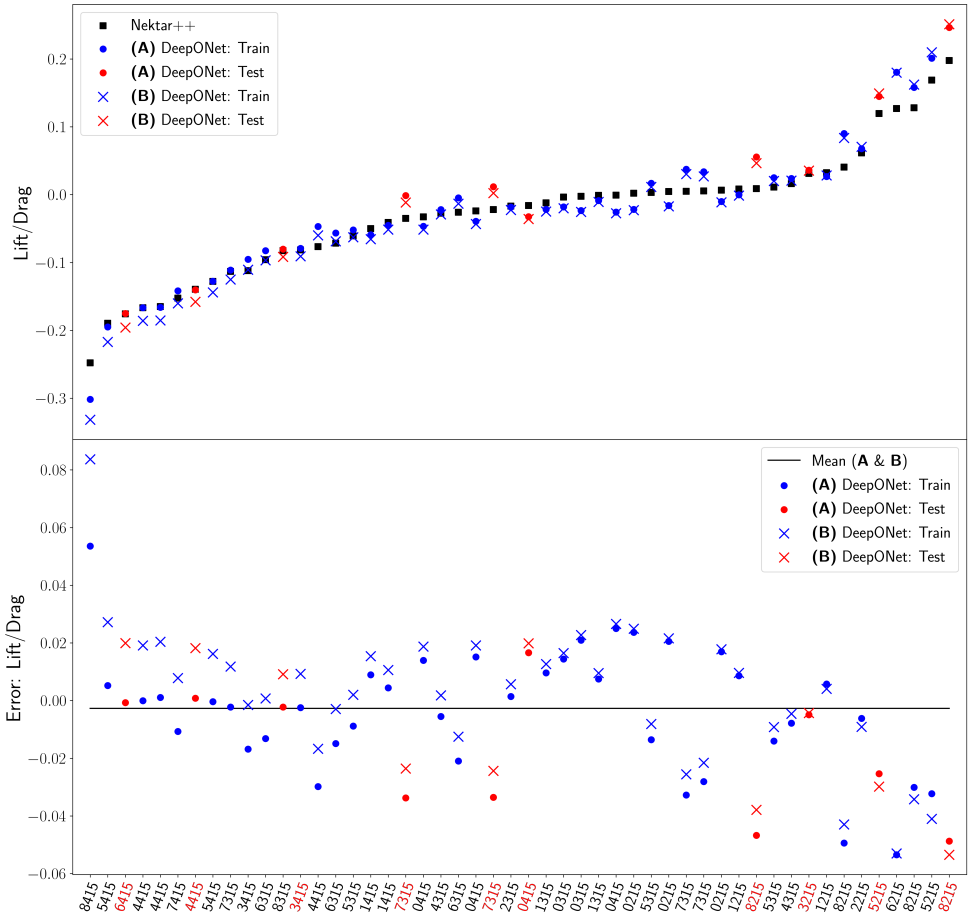


图 9：按 Nektar++ 参考值排序的整个数据集的升阻比目标函数计算结果图。如两幅图所示，在整个参数域内，对高保真计算流体力学解的近似都非常准确且一致。特别是，我们注意到，测试集的表现与训练集相当，这意味着几乎不存在泛化误差；而在优化过程中推断未见过的查询几何形状时，这一点至关重要。

将方程（12）中的随机种子设定为用于拉丁超立方抽样（LHS），以生成构建初始高斯过程所需的初始点集。为提高效率，我们使用 Dakota 提供的批次序贯并行化，在一台八核中央处理器（2.3 GHz 英特尔酷睿 i9）上运行。

算法 1 将基于 DeepONet 的代理模型与 Dakota 集成 要求： m, p, x, y：最大弯度、最大弯度位置、用于流场预测的空间坐标 要求：训练好的 DeepONet： $\mathcal{G}(m, p, x, y)$ 要求： $\mathcal{D}$： $\mathcal{D} \in$ Dakota中的算法 要求： Ψ (u, v, ρ, p, ξ_g)：生成升力 L 和阻力 D的函数 要求： N：目标函数评估次数 要求： $m_{\min}, m_{\max}, p_{min}, p_{\max}$：可行域$n \leftarrow 0$, $m_{\text{opt}} \leftarrow m_0$, $p_{\text{opt}} \leftarrow p_0$的边界

▷ 初始化

当 $n\ !\ =\ N$ **执行**$m_{\text{opt}},\ p_{\text{opt}} \leftarrow \mathcal{D}(-f(m_{\text{opt}},\ p_{\text{opt}})\ m_{\min},\ m_{\max},\ p_{min},\ p_{\max},\ \mathcal{G},\ \Psi)$ ▷ 参数优化过程

$n \leftarrow n + 1$**结束循环**

方法（A）和（B）的优化景观如图 10 所示，这些景观是通过对各自的目标函数进行暴力评估得到的。图中还标出了参数空间（$p,\ m$）$\in [0.2, 0.5] \times [0.0, 0.09]$中的数据集位置，展示了为获得准确优化结果而采用的稀疏采样。我们还观察到，该

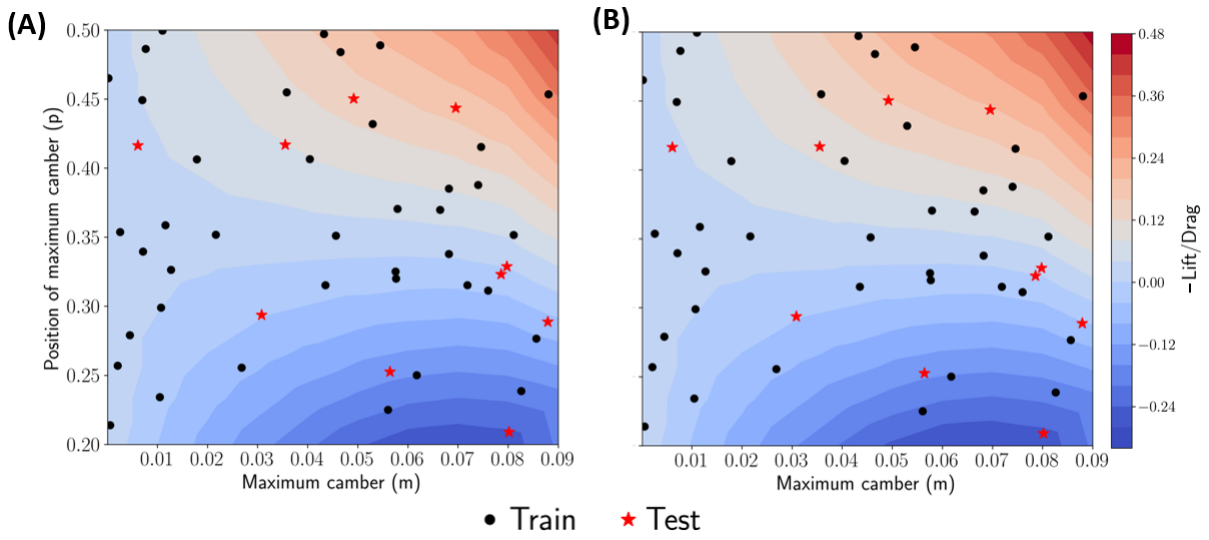


图 10：通过对 p 和 m 进行暴力采样并采用 10×10 网格，得到的升阻比景观可视化。图中还绘出了训练集和测试集，以展示 DeepONet 使用的稀疏数据集。(A) 使用有限差分近似得到的景观。(B) 使用自动微分近似得到的景观。

模型类型	单次目标函数评估的相对成本
基准计算流体力学 (Nektar++placeholder)	32，253
DeepONet (**A**)	1.34
DeepONet（**B**）	1

表4：几何优化过程中单次目标函数评估的相对成本。A：采用有限差分近似的流场映射。B：采用自动微分近似的流场映射。

实验设置简单且为凸问题。在未来的工作中，局部变形等更复杂的条件会使优化景观更加复杂，且很可能变为非凸问题，因此需要更复杂的优化方法。虽然本文并不需要如此，我们仍在框架中使用了最先进的优化方法 Dakota。此外，我们可以看到，在将参数边界限制为数据集采样边界时，局部最小值位于边界处。这意味着全局最小值位于训练边界之外，而生成数据时无法预先得知这一点。未来，我们希望评估 DeepONets 在训练边界之外进行外推的能力，并可能使用迁移学习。

优化过程通过 15 次评估找到了函数的最小值点，而使L/D达到最大值的最优值为（m_*，p_*）=（0.2，0.067）。为确保优化过程的一致性，我们使用不同的随机种子运行了 15 次 EGO算法，并观察到相同的最优点。此外，我们在附录5.2中通过与其他方法进行比较，验证了优化结果。所有方法得到的结果高度一致，详细结果及其实际运行时间列于表6。最后，本文最重要的贡献见表4。我们可以看到，将 DeepONet 集成到翼型几何优化框架中，使新目标函数评估的在线成本降低了 32，000+ 倍。因此，几乎实时地获得优化结果完全可行，所需时间仅为几分钟，而非数天。此外，训练好的模型可以部署在任何硬件上，例如普通笔记本电脑，并能在几秒内预测出准确的实时流场，这意味着测试时的几何优化不受硬件限制。我们证明，将 DeepONets 集成到几何优化流程中，精度损失很小；与传统 CFD 方法相比，这种方法以获取离线数据集并在其上训练为代价，换来了在线使用时几乎即时的优化结果。

为验证优化翼型的参数（$p = 0.2$，0.067），我们比较了图 11 中的流线图。这些流线图根据训练好的 DeepONet 和 Nektar++ 得到的流场（u，v）绘制而成。总体而言，流线图显示出极佳的一致性，因此验证了本文提出的形状优化工作流程。DeepONet 能够成功检测并预测翼型后缘附近的环量。然而，仔细观察会发现，有些非物理流线起始于翼型表面。这是由于 DeepONet 预测的翼型表面附近速度场存在误差。训练 DeepONet 时，若通过适当加权提高翼型表面附近区域的重要性，结果还可以进一步改善。

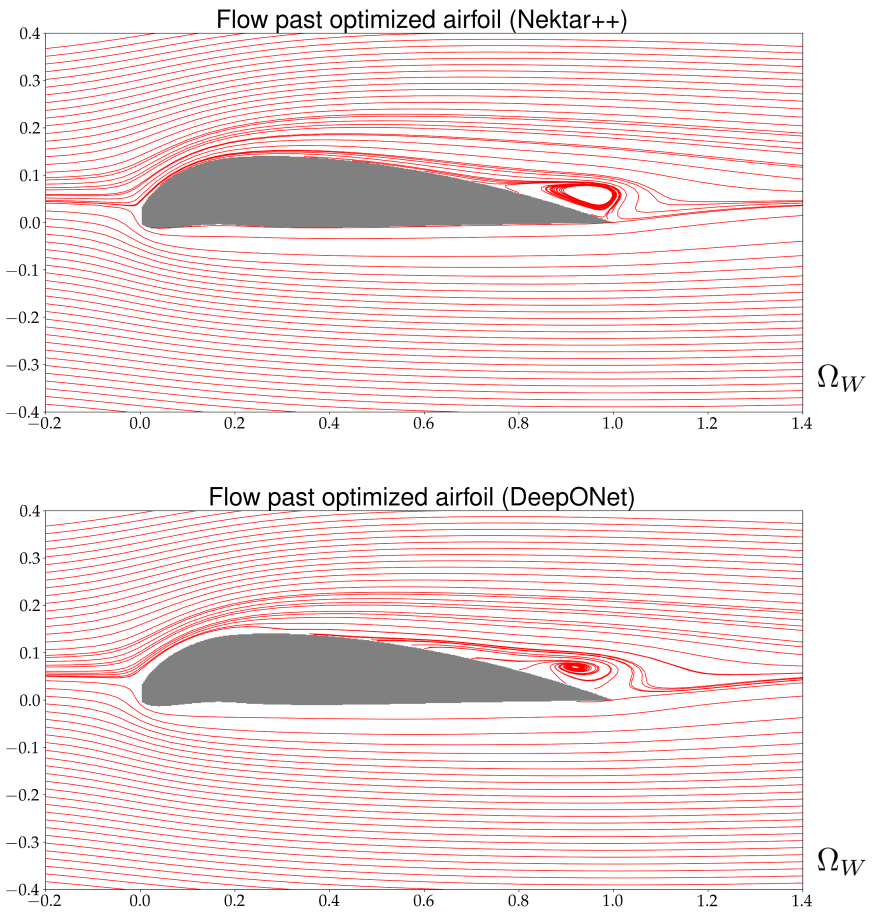


图 11：**优化翼型**。流线图展示了参数为（p，m）=(0.2，0.067) 的优化翼型周围流动，区域为 Ω_W。此处展示了在 Nektar++ 中模拟的速度场以及 DeepONets 预测的速度场。

5 结论

我们已成功将 DeepONets 作为代理模型集成到翼型形状优化框架中。在总结该领域已有工作的基础上，我们通过实证展示了 DeepONets 的有效性：其流场精度足以用于评估升阻比目标函数，同时在在线几何优化中取代传统计算流体力学求解器，可显著提升计算速度。我们对所呈现的结果进行了全面验证，并开展了广泛的实验，包括近似壁面剪切应力的两种方法（A）和（B），以及两种 DeepONet 输入形式（非均匀有理B样条和 ξ_g），以确保流程稳健。值得注意的是，DeepONets 在整个数据集上几乎没有泛化误差，因此得到的优化几何形状（$p = 0.2$，$m = 0.067$）是准确的；与计算流体力学基准相比，速度提升了 32，253 倍。该框架具有通用性，可处理包含多个输入的更复杂问题，例如不同的马赫数和攻角，这些参数可作为分支网络或主干网络的输入。因此，只需进行相对少量的修改，该框架即可用于具有流动非定常性、激波、非平衡化学甚至变形几何的高速流动状态下的优化。此外，由于采用 DeepONet 将机器学习集成到函数到函数映射中，本文提出的方法具有灵活性。因此，可利用多保真度训练和物理信息机器学习等改进方法降低数据生成成本。我们还成功展示了自动微分的应用；在壁面剪切应力计算中，其表现与传统有限差分方法相当。最后，我们希望利用近期开发的库 *NeuralUQ* [59] 进行迁移学习和不确定性量化，从而在训练所覆盖的几何参数域之外进行外推，并有把握地找到全局最优解。

致谢

本文所报告的研究是与美国陆军研究实验室合作协议合同/文书 W911NF-22-2-0047 相关开展的。L.B. 和 A.G. 获得了美国陆军研究实验室车辆动力与推进科学6.1基础研究计划的支持。本文所述观点和结论仅代表作者观点，不应被解读为美国陆军研究实验室或美国政府明示或暗示的官方政策或立场。尽管本文中注明了版权信息，美国政府仍有权为政府用途复制和分发重印本。

附录

5.1 Nektar++交叉验证

为确保得到稳态解，我们选择流动参数为 $M = 0.5$ 和 $Re = 500$。计算稳态解需要仔细考量。为确保流动达到稳态，对于每个 NACA翼型，我们记录了尾迹中六个不同位置处守恒变量的数值。随后，我们检查守恒变量的时间历程，以确认系统已达到稳态且解的更新已停止。图12显示了这些历史数据采样点的位置，这些位置处的瞬态流动可能比其他空间位置持续更久。作为示例，我们绘制了点5处变量的时间历程（图13）；该点的流动波动幅度随时间变化最大。根据图13，当守恒变量趋于恒定值时，解达到了稳态。

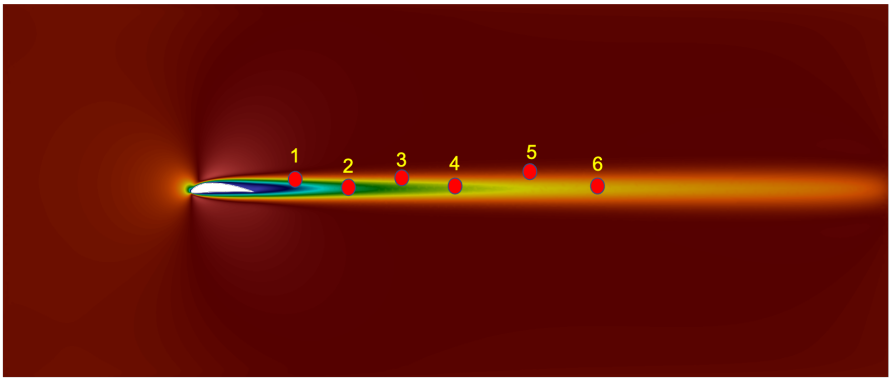


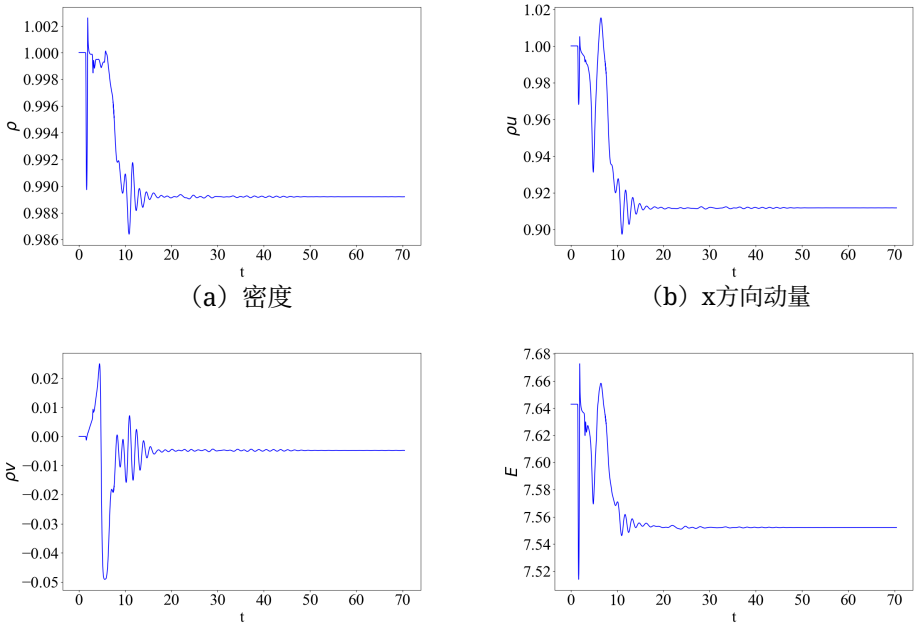
图12：NACA翼型尾迹中的历史点位置。在这些位置随时间存储守恒变量值，以监测稳态解。

我们还将 NekTar++ 中 NACA翼型的仿真设置，与基于 Kopriva [60, 61] 的间断谱元方法开发的内部代码所得结果进行了验证。我们选择 NACA0020翼型进行交叉验证。计算了采用 $M = 0.5$ 和 $Re = 500$ 的流动稳态解，结果如图14所示。两种求解器计算得到的流场原始变量一致，验证了 NekTar++ 仿真设置的有效性，包括网格和仿真参数。根据图14，NekTar++ 仿真所采用的单元数量足以进行准确预测。在验证流场后，我们还对 NACA4402翼型周围的流动进行了仿真。Kunz [62] 报告了该翼型的阻力系数和升力系数，但其对应的是 $Re = 1000$下的不可压缩流。

我们采用训练集所使用的自动网格生成设置来创建网格，并使用与训练集相同的仿真设置。我们计算了阻力系数和升力系数，并将其与文献中的结果进行了比较。表5比较了由 Nektar++ 计算得到的阻力系数和升力系数与 Kunz [62] 报告的、针对相似但并非完全相同设置的数值。

	C_d	C_l	误差
Nektar++	0.1050	0.1700	1.7%
Kunz [62]	0.1032	0.1852	8.2%

表5：使用 Nektar++ 计算的阻力系数和升力系数与使用 Kunz [62] 计算的结果之间的比较。对于参考解，采用了不同的设置，即雷诺数为 1000，且假设流动不可压缩。



（a）密度　（b）x方向动量

（c）y方向动量　（d）总能量

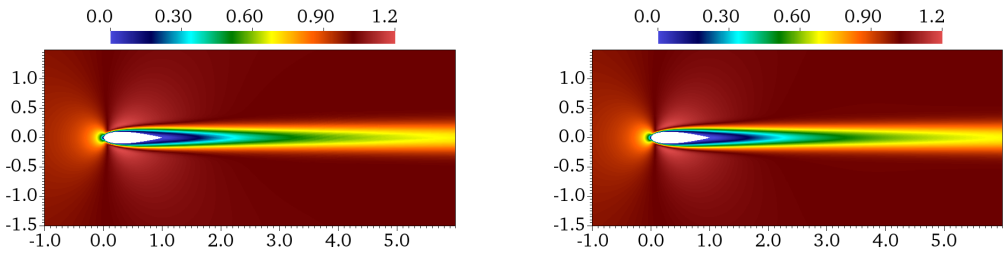
图13：NACA7315翼型在点5（见图12）处（a） ρ、（b） ρu、（c） ρv和（d） E 的时间历程。曲线表明，模拟已达到稳态解，流动变量达到恒定状态。

模型类型	优化参数 (p, m)	优化后的 $\frac{L}{D}$	计算成本（秒）
暴力搜索（**A**）	(0.2, 0.070)	0.272	332.63
暴力搜索（**B**）	(0.2, 0.070)	0.281	264.12
SciPy 双重退火（**A**）	(0.2, 0.064)	0.269	196.09
SciPy 双重退火（**B**）	(0.2, 0.064)	0.281	165.57
Dakota（**A**）	(0.2, 0.067)	0.269	157.71
Dakota（**B**）	(0.2, 0.063)	0.282	105.00

表6：不同 DeepONet 方法的几何优化结果。(A) 使用有限差分近似进行流场映射。(B) 使用自动微分近似进行流场映射。这些方法的计算成本是在 MacBook Pro 2019 的八核中央处理器（2.3 GHz 英特尔酷睿 i9）上测得的，该设备配备 16 GB 2667 MHz DDR4 内存。

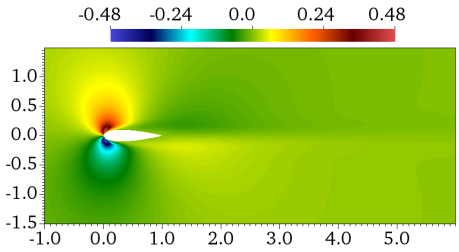
5.2 几何优化验证

为了使用 Dakota 验证论文中的主要几何优化结果，我们还分别采用暴力搜索和 SciPy 的 [63] 双重退火法计算了目标函数。暴力搜索在几何参数空间（p， m）上使用 10×10 网格进行评估 $\in [0.2, 0.5] \times [0.0, 0.09]$，因此需要对目标函数进行100次评估。双重退火法的最大评估次数设为50次，但很可能可以设置得更少。如表6所示，无论采用哪种优化器，或使用由（A）和（B）给出的哪种部分近似，所有方法都找到了相同的最优参数集和 $\frac{L}{D}$。

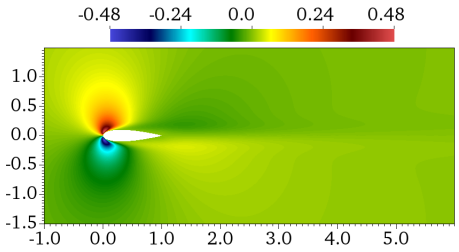


(a) x 方向速度 Nektar++

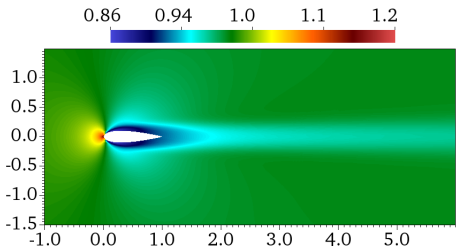
(b) x 方向速度 DSEM



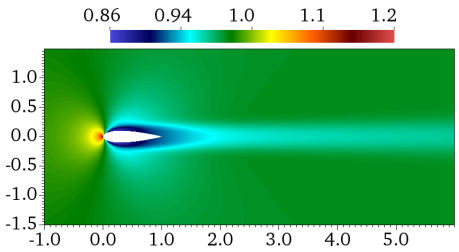
(c) y 方向速度 Nektar++



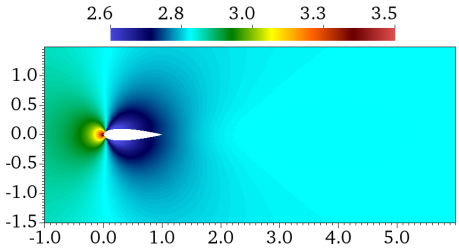
(d) y 方向速度 DSEM



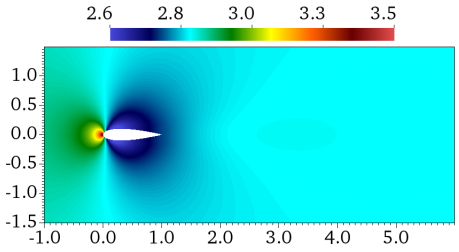
(e) 密度 Nektar++



(f) 密度 DSEM



(g) 压力 Nektar++



(h) 压力 DSEM

图 14：对比验证使用 DSEM 代码和 Nektar++ 得到的绕 NACA0020 稳态流动。流动参数设置为 $Re = 500$和 $M = 0.5$。

参考文献

[1] Lu Lu、Pengzhan Jin、Guofei Pang、Zhongqiang Zhang 和 George Em Karniadakis。基于算子的通用逼近定理，通过 DeepONet 学习非线性算子。*Nature Machine Intelligence*，3(3):218–229，2021年。[2] Lu Lu、Pengzhan Jin 和 George Em Karniadakis。DeepONet：基于算子的通用逼近定理，通过学习非线性算子识别微分方程。*arXiv* 预印本 *arXiv:1910.03193*，2019年。[3] Maziar Raissi、Paris Perdikaris 和 George E 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逆问题的深度学习框架。计算物理学杂志，378:686–707，2019年。[4] Subhayan De、Malik Hassanaly、Matthew Reynolds、Ryan N. King 和 Alireza Doostan。利用 DeepONet 对不确定且部分未知的系统进行双保真建模。预印本，网址：https://arxiv.org/abs/2204.00997，2022年。[5] Amanda A. Howard、Mauro Perego、George E. Karniadakis 和 Panos Stinis。多保真深度算子网络。预印本，网址：https://arxiv.org/abs/2204.09157，2022年。[6] Lu Lu、Raphaël Pestourie、Steven G. Johnson 和 Giuseppe Romano。多保真深度神经算子：高效学习偏微分方程，并应用于纳米尺度热传输的快速逆向设计。物理评论研究，4:023210，2022年。[7] Pengzhan Jin、Shuai Meng 和 Lu Lu。MIONet：通过张量积学习多输入算子。*arXiv* 预印本 *arXiv:2202.06137*，2022年。[8] Min Zhu、Handi Zhang、Anran Jiao、George Em Karniadakis 和 Lu Lu。由物理信息或稀疏观测指导的深度神经算子的可靠外推。*arXiv* 预印本 *arXiv:2212.06347*，2022年。[9] Jan S Hesthaven 和 Stefano Ubbiali。利用神经网络对非线性问题进行非侵入式降阶建模。计算物理学杂志，363:55–78，2018年。[10] Jan S Hesthaven、Gianluigi Rozza、Benjamin Stamm 等。《参数化偏微分方程的认证降阶基方法》，第590卷。Springer，2016年。[11] Peter Benner、Mario Ohlberger、Anthony Patera、Gianluigi Rozza 和 Karsten Urban。《参数化系统的模型降阶》。Springer，2017年。[12] Matthew O Williams、Ioannis G Kevrekidis 和 Clarence W Rowley。Koopman算子的数据驱动近似：扩展动态模态分解。非线性科学杂志，25(6):1307–1346，2015年。[13] Eliodoro Chiavazzo、Charles W Gear、Carmeline J Dsilva、Neta Rabin 和 Ioannis G Kevrekidis。通过非线性数据挖掘构建化学动力学降阶模型。*Processes*，2(1):112–140，2014年。[14] Chad Lieberman、Karen Willcox 和 Omar Ghattas。面向大规模统计反问题的参数与状态模型降阶。*SIAM*科学计算杂志，32(5):2523–2542，2010年。[15] Tan Bui-Thanh、Karen Willcox 和 Omar Ghattas。针对高维参数输入空间的大规模系统模型降阶。*SIAM*科学计算杂志，30(6):3270–3288，2008年。[16] Peter Benner、Serkan Gugercin 和 Karen Willcox。参数化动力系统基于投影的模型降阶方法综述。*SIAM*评论，57(4):483–531，2015年。[17] David Amsallem、Matthew Zahr、Youngsoo Choi 和 Charbel Farhat。利用超降阶模型进行设计优化。结构与多学科优化，51(4):919–940，2015年。[18] Kevin Carlberg 和 Charbel Farhat。用于面向优化的降阶模型的紧凑本征正交分解基。载于第*12*届*AIAA/ISSMO*多学科分析与优化会议，第5964页，2008年。[19] Youngsoo Choi、Gabriele Boncoraglio、Spenser Anderson、David Amsallem 和 Charbel Farhat。利用线性降阶模型数据库进行基于梯度的约束优化。计算物理学杂志，423:109787，2020年。[20] Katiana Kontolati、Somdatta Goswami、Michael D Shields 和 George Em Karniadakis。过参数化对基于流形的代理模型和深度神经算子的影响。*arXiv* 预印本 *arXiv:2203.05071*，2022年。[21] Yin Yu、Zhoujie Lyu、Zelu Xu 和 Joaquim RRA Martins。优化算法和初始设计对机翼气动外形优化的影响。航空航天科学与技术，75:183–199，2018年。

[22] James Reuther、Antony Jameson、James Farmer、Luigi Martinelli 和 David Saunders。通过伴随公式对复杂飞机构型进行气动外形优化。载于第34届航空航天科学会议及展览，第94页，1996年。[23] Giampietro Carpentieri、Barry Koren 和 Michel JL van Tooren。非结构网格上的基于伴随方法的气动外形优化。《计算物理学杂志》，224(1):267–287，2007年。[24] Siva Nadarajah 和 Antony Jameson。黏性流自动气动外形优化中连续伴随方法与离散伴随方法的研究。载于第15届AIAA计算流体力学会议，第2530页，2001年。[25] DN Srinath 和 Sanjay Mittal。非定常黏性流形状优化的伴随方法。《计算物理学杂志》，229(6):1994–2008，2010年。[26] Oleg Chernukhin 和 David W Zingg。气动设计中的多模态性与全局优化。《AIAA期刊》，51(6):1342–1354，2013年。[27] Jichao Li、Jinsheng Cai 和 Kun Qu。利用主动子空间方法进行基于代理模型的气动外形优化。《结构与多学科优化》，59(2):403–419，2019年。[28] Xiaojing Wu、Zijun Zuo 和 Long Ma。面向受约束跨声速翼型与机翼外形设计的气动数据驱动代理辅助教学优化（TLBO）框架。《Aerospace》，9(10):610，2022年。[29] Russell Eberhart 和 James Kennedy。一种采用粒子群理论的新型优化器。载于 MHS'95：《第六届微型机器与人类科学国际研讨会论文集》，第39–43页。IEEE，1995年。[30] Daniel G Krige。关于威特沃特斯兰德若干基本矿山估值问题的统计方法。《南非矿业与冶金学会期刊》，52(6):119–139，1951年。[31] J Liu、W-P Song、Z-H Han 和 Y Zhang。利用并行填充策略和代理模型高效优化跨声速机翼气动外形。《结构与多学科优化》，55(3):925–943，2017年。[32] Jerome Lepine、Francois Guibault、Jean-Yves Trepanier 和 Francois Pepin。用于机翼气动设计的优化非均匀有理B样条几何表示。《AIAA期刊》，39(11):2033–2041，2001年。[33] Kun Wang、Shengjiao Yu、Zheng Wang、Renzhong Feng 和 Tiegang Liu。采用自适应等几何不连续伽辽金方法进行基于伴随方法的翼型优化。《计算力学与工程方法》，344:602–625，2019年。[34] Dimitrios I Papadimitriou 和 Costas Papadimitriou。面向最小鲁棒阻力和升力可靠性约束的气动外形优化。《航空航天科学与技术》，55:24–33，2016年。[35] Raymond M Hicks 和 Preston A Henne。通过数值优化进行机翼设计。《飞机杂志》，15(7):407–412，1978年。[36] Simon Painchaud-Ouellet、Christophe Tribes、Jean-Yves Trépanier 和 Dominique Pelletier。在厚度约束下采用非均匀有理B样条参数化进行翼型形状优化。《AIAA期刊》，44(10):2170–2178，2006年。[37] Guodong Chen 和 Krzysztof Fidkowski。利用基于输出的自适应网格进行翼型形状优化。载于第23届AIAA计算流体力学会议，第3102页，2017年。[38] Xiaolong He、Jichao Li、Charles A Mader、Anil Yildirim 和 Joaquim RRA Martins。鲁棒气动外形优化——从圆形到翼型。《航空航天科学与技术》，87:48–61，2019年。[39] Xiaojing Wu、Weiwei Zhang、Xuhao Peng 和 Ziyi Wang。采用基于本征正交分解的CST翼型参数化方法进行基准气动外形优化。《航空航天科学与技术》，84:632–640，2019年。[40] Md Tausif Akram 和 Man-Hoe Kim。利用类别函数/形状函数变换和遗传算法进行翼型CFD分析与形状优化——第一部分。《应用科学》，11(9):3791，2021年。[41] Xinshuai Zhang、Fangfang Xie、Tingwei Ji、Zaoxu Zhu 和 Yao Zheng。用于气动外形优化的多保真度深度神经网络代理模型。《计算力学与工程方法》，373:113485，2021年。[42] SUN Zhiwei、WANG Chen、Yu Zheng、BAI Junqiang、LI Zheng、XIA Qiang 和 FU Qiujun。用于预测变化几何形状下跨声速流的非侵入式降阶模型。《中国航空学报》，33(2):508–519，2020年。[43] S Ashwin Renganathan、Romit Maulik 和 Jai Ahuja。结合深度神经网络和高斯过程提高气动设计优化的数据效率。《航空航天科学与技术》，111:106522，2021年。

[44] Xiaosong Du、Ping He 和 Joaquim RRA Martins。基于神经网络参数化和代理建模的快速翼型设计优化。航空航天科学与技术，113:106701，2021年。[45] Peng Liao、Wei Song、Peng Du 和 Hang Zhao。基于迁移学习的多保真卷积神经网络代理模型在气动优化中的应用。流体物理，33(12):127121，2021年。[46] Jun Tao 和 Gang Sun。基于深度学习的多保真代理模型在鲁棒气动设计优化中的应用。航空航天科学与技术，92:722–737，2019年。[47] Christophe Geuzaine 和 Jean-Francois Remacle。Gmsh。[48] C.D. Cantwell、D. Moxey、A. Comerford、A. Bolis、G. Rocco、G. Mengaldo、D. De Grazia、S. Yakovlev、J.-E. Lombard、D. Ekelschot、B. Jordi、H. Xu、Y. Mohamied、C. Eskilsson、B. Nelson、P. Vos、C. Biotto、R.M. Kirby 和 S.J. Sherwin。Nektar++：一个开源谱/hp 单元框架。计算机物理通讯，192:205–219，2015年。[49] David Moxey、Chris D. Cantwell、Yan Bao、Andrea Cassinelli、Giacomo Castiglioni、Sehun Chun、Emilia Juda、Ehsan Kazemi、Kilian Lackhove、Julian Marcon、Gianmarco Mengaldo、Douglas Serson、Michael Turner、Hui Xu、Joaquim Peiró、Robert M. Kirby 和 Spencer J. Sherwin。Nektar++：提升高保真谱/hp 单元方法的能力与应用。计算机物理通讯，249:107110，2020年。[50] Eastman N Jacobs、Kenneth E Ward 和 Robert M Pinkerton。变密度风洞测试中 78 个相关翼型剖面的特性。美国国家航空咨询委员会，1933年。[51] Onur Rauf Bingol 和 Adarsh Krishnamurthy。NURBS-Python：一个基于 Python 的开源面向对象 NURBS 建模框架。*SoftwareX*，9:85–94，2019年。[52] de Berg Mark、Cheong Otfried、van Kreveld Marc 和 Overmars Mark。计算几何算法及其应用。Spinger，2008年。[53] Gianmarco Mengaldo、Daniele De Grazia、Freddie Witherden、Antony Farrington、Peter Vincent、Spencer Sherwin 和 Joaquim Peiro。可压缩空气动力学中紧致高阶方法边界条件实现指南。载于第七届 *AIAA* 理论流体力学会议，第 2923 页，2014年。[54] Tianping Chen 和 Hong Chen。具有任意激活函数的神经网络对非线性算子的通用逼近及其在动力系统中的应用。*IEEE* 神经网络汇刊，6(4):911–917，1995年。[55] Zongyi Li、Nikola Kovachki、Kamyar Azizzadenesheli、Burigede Liu、Kaushik Bhattacharya、Andrew Stuart 和 Anima Anandkumar。神经算子：用于偏微分方程的图核网络，2020年。[56] Atilim Gunes Baydin、Barak A Pearlmutter、Alexey Andreyevich Radul 和 Jeffrey Mark Siskind。机器学习中的自动微分：综述。机器学习研究杂志，18:1–43，2018年。[57] Brian Adams、William Bohnhoff、Keith Dalbey、Mohamed Ebeida、John Eddy、Michael Eldred、Russell Hooper、Patricia Hough、Kenneth Hu、John Jakeman、Mohammad Khalil、Kathryn Maupin、Jason A. Monschke、Elliott Ridgway、Ahmad Rushdi、Daniel Seidl、John Stephens、Laura Painton Swiler、Anh Tran 和 Justin Winokur。Dakota：用于设计优化、参数估计、不确定性量化和敏感性分析的多层并行面向对象框架：6.16 版用户手册。[58] D Finkel 和 Carl Tim Kelley。直接算法的收敛性分析。技术报告，北卡罗来纳州立大学科学计算研究中心，2004年。[59] Zongren Zou、Xuhui Meng、Apostolos F Psaros 和 George Em Karniadakis。NeuralUQ：用于神经微分方程和算子不确定性量化的综合库。*arXiv* 预印本 *arXiv:2208.11866*，2022年。[60] David A. Kopriva 和 John H. Kolias。用于可压缩流的守恒型交错网格切比雪夫多区域方法。计算物理学杂志，125(1):244–261，1996年。[61] Ahmad Peyvan、Jonathan Komperda、Dongru Li、Zia Ghiasi 和 Farzad Mashayek。在间断谱元方法中使用雅可比修正函数进行通量重构。计算物理学杂志，435:110261，2021年。[62] Peter Josef Kunz。超低雷诺数飞行的空气动力学与设计。斯坦福大学，2003年。[63] Pauli Virtanen、Ralf Gommers、Travis E. Oliphant、Matt Haberland、Tyler Reddy、David Cournapeau、Evgeni Burovski、Pearu Peterson、Warren Weckesser、Jonathan Bright、Stéfan J. van der Walt、Matthew Brett、Joshua Wilson、K. Jarrod Millman、Nikolay Mayorov、Andrew R. J. Nelson、Eric Jones、Robert Kern、Eric Larson、C J Carey、Ilhan Polat、Yu Feng、Eric W. Moore、Jake VanderPlas、Denis Laxalde、Josef Perktold、Robert Cimrman、Ian Henriksen、E. A. Quintero、Charles R. Harris、Anne M. Archibald、Antônio H. Ribeiro、Fabian Pedregosa、Paul van Mulbregt 和 SciPy 1.0 贡献者。SciPy 1.0：Python 科学计算的基础算法。自然·方法，17:261–272，2020年。