有效动力学的自适应学习：复杂系统的自适应实时在线建模

Ivica Kičić[a]、Pantelis R. Vlachas[a,b]、Georgios Arampatzis[a,b]、
MichailChatzimanolakis[a,b]、Leonidas Guibas、Petros Koumoutsakos[b,*]

[a]计算科学与工程实验室，苏黎世联邦理工学院，瑞士 *CH-8092*[b]工程与应用科学学院，牛津街 *29* 号，哈佛大学，美国马萨诸塞州剑桥市，邮编 *MA02138*[c]计算机科学系，塞拉楼 *353* 室，斯坦福大学，美国加利福尼亚州，邮编 *CA 94035*

arXiv:2304.01732v1 [physics.comp-ph] 4 Apr 2023

摘要

预测模拟对于从天气预报到材料设计等各种应用都至关重要。这些模拟的真实性取决于其捕捉有效系统动力学的能力。大规模并行模拟通过解析所有时空尺度来预测系统动力学，但其计算成本往往高到无法进行实验。另一方面，降阶模型虽然速度较快，但通常受限于系统动力学的线性化以及所采用的启发式闭合方法。我们提出了一种新颖的系统性框架，将大规模模拟与降阶模型相结合，以自适应地提取和预测多尺度系统的有效动力学（AdaLED）。AdaLED 采用自编码器来识别系统动力学的降阶表示，并使用概率循环神经网络（RNNs）的集成作为潜在时间步进器。该框架在计算求解器与代理模型之间交替运行，在加速已学习动力学的同时，将尚未学习的动力学状态区间交由原始求解器处理。AdaLED 通过在线训练持续使代理模型适应新的动力学。通过监测代理模型的预测精度和不确定性，确定代理模型与计算求解器之间的切换时机。我们在三个不同的系统上验证了 AdaLED 的有效性：范德波尔振子、二维反应扩散方程，以及雷诺数变化范围为 400 至 1200 的二维圆柱绕流纳维-斯托克斯流动。这些实验展示了其在线学习有效动力学、检测未见过的动力学状态区间以及实现净加速的能力。据我们所知，AdaLED 是首个将代理模型与计算求解器相结合，以实现有效动力学在线自适应学习的框架。它为需要大量计算成本高昂的模拟的应用提供了一种强有力的工具。

关键词： 自适应降阶建模、计算机模拟、机器学习、在线实时学习、连续学习、纳维-斯托克斯方程

1. 引言

复杂系统的模拟提升了我们的预测能力，其应用领域涵盖健康与流行病学 [1] 、物理学 [2], 、气象学 [3], 和流体力学。在天体物理学和气候科学等难以开展实验的领域，或实验需要昂贵的基础设施、设备和人员的领域，大规模模拟尤为重要。

模拟的预测保真度取决于其解析所研究物理现象相关时空尺度的能力。然而，高保真度意味着高昂的计算成本，

*通讯作者电子邮箱： kicici@ethz.ch (Ivica Kičić), pvlachas@ethz.ch (Pantelis R. Vlachas), garampat@ethz.ch (Georgios Arampatzis), michaich@ethz.ch (Michail Chatzimanolakis),guibas@cs.stanford.edu (Leonidas Guibas), petros@seas.harvard.edu (Petros Koumoutsakos)

这阻碍了实验和优化。许多科学与工程任务，例如参数优化与设计优化 [4], 多目标优化 [5], 强化学习（RL） [6, 7, 8], 以及高通量计算 [9], ，都需要对系统进行大量评估。当为这些任务使用高精度模拟固然理想，但成本可能高得难以承受，甚至难以实施。因此，众多研究致力于开发准确且高效的代理模型，以复制并加速模拟。

尽管计算成本高昂的模拟对于解析复杂系统的所有尺度至关重要，但关键关注量通常可以用粗粒化的平均行为来描述。如何为此类粗粒化表征选择合适的自由度，是科学与工程领域长期存在的问题。此外，恰当结合粗粒化模拟、细尺度模拟与预测，有望在准确度可控的情况下加速模拟。早期的混合方法包括无方程框架（EFF） [10, 11, 12], 、异构多尺度方法（HMM） [13, 14],和 FLow AVeraged integratoR（FLAVOR） [15]。混合方法区分了成本高昂的高维物理空间（微观尺度）与粗粒化、降阶或潜在空间（宏观尺度）。具体而言，在 EFF 中，系统首先在成本高昂的微观尺度上推进一段给定时间。随后，借助主成分分析、动态模态分解 [16], 或扩散映射 [17]等压缩机制，转换至宏观尺度。接着，EFF 采用欧拉方法或龙格-库塔方法等时间步进格式推进宏观尺度动力学。经过若干时间步后，再将宏观尺度动力学映射回细尺度，进行详细模拟。通过在关注的时间尺度上交替推进微观尺度与宏观尺度动力学，EFF 能显著节省计算成本。然而，由于微观尺度与宏观尺度之间的信息传递不够恰当，且宏观尺度传播器效率低下，EFF 在复杂高维系统中的泛化能力受到限制。

近年来，人们开展了大量工作，致力于利用机器学习（ML） [18, 19, 20, 21, 22, 23]开发降阶模型并加速复杂模拟。在以往的工作中，我们扩展了 EFF，采用机器学习算法以数据驱动的方式学习时间积分器和传递算子。由此形成的复杂动力系统有效动力学学习（LED） [24] 框架，使用卷积自编码器（CAEs）辨识微观到宏观及宏观到微观的映射，并使用循环神经网络（RNNs）传播宏观动力学。自编码器和 RNN 使用来自微观传播器（即原始模拟器）的数据进行离线训练。LED 已应用于多种动力系统 [25],，涵盖从流体流动到分子模拟 [26]的领域。

值得注意的是，与 LED 相关的框架包括偏微分方程的潜在演化（LE-PDE） [27]。与此同时，卷积自编码器（CAEs）与长短期记忆网络（LSTMs）相结合，已被用于复杂流动建模 [28, 29, 30, 31, 32, 33, 34]。其他与 LSTM 网络结合的自编码器（AEs）也被用于高维动力系统的代理建模 [35],，例如圆柱体绕流的非定常流动 [36, 37], ，或两层建筑的结构建模 [38]。其他值得关注的工作基于本征正交分解（POD） [39, 40, 41, 42, 43], 、采用 POD 的局部近似 [44, 45], 、动态模态分解（DMD） [16], 以及动力学辨识（ID） [46, 47]。文献 [48, 49]提出了采用低秩更新的自适应扩展方法。

然而，据我们所知，上述框架并不具备以下一项或多项特征：

1. 它们缺乏连续训练，因此难以适应变化的动力学或进行泛化 e
对于初始训练数据中代表性不足的区域。2. 它们不量化预测不确定性，也不监测预测误差。3. 它们忽视了只有在可靠时才应使用代理模型。4. 它们没有利用从任意时间点重新启动计算机仿真的能力。5. 它们无法控制加速比与准确度之间的平衡。6. 它们没有考虑细尺度动力学参数的变化。

尽管已有多项研究提出了稳健的代理建模框架，但实时应用以及涉及非定常动力学或状态空间探索的任务，需要采用持续学习框架来解决分布偏移问题 [50]。

代理模型构建中的自适应性和持续学习是本文的关键组成部分。本文提出的方法 有效动力学的自适应学习（AdaLED）通过在线学习代理模型来加速计算成本高昂的模拟；只有当其预测足够可靠时，才用该模型替代原始模拟器。具体而言，AdaLED 在 LED 框架中加入了不确定性量化。代理模型会监测其预测精度，并为预测提供置信度。相应地，代理模型仅用于其已经学习并且确信预测结果的状态空间区域（轨迹的一部分）。否则，则使用计算求解器模拟代理模型尚不熟悉的动力学。最后，我们为 LED 扩展了持续学习能力，以应对动力学随时间变化的环境/动力系统中的分布偏移问题。

我们证明，AdaLED 能够自适应地学习复杂动力学，生成可靠的代理模型预测，并在保持高准确度的同时加速计算成本高昂的模拟。AdaLED 通过自适应地设定模拟的误差阈值，在准确度与速度之间实现可控权衡。

本文结构如下：第 2 节详细介绍 AdaLED 框架；第 3 节和第 4 节分别展示 AdaLED 在范德波尔振子和二维反应扩散系统中的效率与效能。第 5 节展示 AdaLED 如何加速不同雷诺数下的二维纳维–斯托克斯圆柱绕流模拟。第 6 节总结全文。关于神经网络以及如何处理超高维流体流动状态的技术信息见附录。

2. 方法

我们考虑微观/精细尺度上动力系统的演化，其状态记为 $\mathbf{x}_t \in \mathbb{R}^{d_x}$，时刻为 t。状态通过 δt 使用微观传播器 $\mathcal{F}$ 进行推进，因此

$$\mathbf{x}_{t+\delta t} = \mathcal{F}(\mathbf{x}_t, \mathbf{f}_t). \tag{1}$$

其中，$\mathbf{f}_t \in \mathbb{R}^{d_f}$ 表示影响动力学的时变外部强迫。此类系统动力学的例子包括湍流的直接数值模拟，以及材料过程中的分子动力学。模拟可提供以$\mathbf{q}_t \in \mathbb{R}^{d_q}$、$\mathbf{q}_t = \mathcal{Q}(\mathbf{x}_t)$表示的关注量。我们假设，有效系统动力学可以用低维潜在状态$\mathbf{z}_t \in \mathcal{M}_z$近似，其中 $\mathcal{M}_z \in \mathbb{R}^{d_z}$ （且 $d_z << d_x$）是系统状态空间中的低阶流形。在此，我们通过采用编码器 $\mathcal{E}^{\boldsymbol{\theta}_\mathcal{E}} : \mathbb{R}^{d_x} \to \mathbb{R}^{d_z}$ 来识别潜在空间，该编码器具有可训练参数 $\boldsymbol{\theta}_\mathcal{E}$。编码器将微观状态$\mathbf{x}_t$ 映射为潜在（宏观）状态$\mathbf{z}_t = \mathcal{E}^{\boldsymbol{\theta}_\mathcal{E}}(\mathbf{x}_t)$。反过来，解码器$\mathcal{D}^{\boldsymbol{\theta}_\mathcal{D}} : \mathbb{R}^{d_z} \to \mathbb{R}^{d_x}$具有可训练参数 $\boldsymbol{\theta}_\mathcal{D}$，可将潜在状态$\mathbf{z}_t$ 映射为微观状态 $\tilde{}\mathbf{x}_t = \mathcal{D}^{\boldsymbol{\theta}_\mathcal{D}}(\mathbf{z}_t)$。最优参数 $\boldsymbol{\theta}^*_\mathcal{E}$ 和 $\boldsymbol{\theta}^*_\mathcal{D}$ 使应用特定的重构损失 $\ell(\mathbf{x}_t, \tilde{}\mathbf{x}_t)$最小：

$$(\boldsymbol{\theta}^*_\mathcal{E}, \boldsymbol{\theta}^*_\mathcal{D}) = \underset{\boldsymbol{\theta}_\mathcal{E}, \boldsymbol{\theta}_\mathcal{D}}{\arg\min}\, \ell\left(\mathbf{x}_t, \tilde{\mathbf{x}}_t\right) = \underset{\boldsymbol{\theta}_\mathcal{E}, \boldsymbol{\theta}_\mathcal{D}}{\arg\min}\, \ell\left(\mathbf{x}_t, \mathcal{D}^{\boldsymbol{\theta}_\mathcal{D}}(\mathcal{E}^{\boldsymbol{\theta}_\mathcal{E}}(\mathbf{x}_t))\right). \tag{2}$$

一个非线性宏观传播器（$\mathcal{H}^{\boldsymbol{\theta}_M}$，$\mathcal{Z}^{\boldsymbol{\theta}_M}$，$\mathcal{Q}^{\boldsymbol{\theta}_M}$，$\mathcal{S}^{\boldsymbol{\theta}_M}$），其参数为 $\boldsymbol{\theta}_M$，并具有一个用于捕捉非马尔可夫效应的内部隐状态$\mathbf{h}_t$，经过训练后可预测宏观尺度上的系统动力学：

$$\mathbf{h}_{t+\Delta t} = \mathcal{H}^{\boldsymbol{\theta}_M}(\mathbf{z}_t, \mathbf{q}_t, \mathbf{f}_t, \mathbf{h}_t), \quad \tilde{\mathbf{z}}_{t+\Delta t} = \mathbf{z}_t + \mathcal{Z}^{\boldsymbol{\theta}_M}(\mathbf{h}_{t+\Delta t}), \quad \tilde{\mathbf{q}}_{t+\Delta t} = \mathbf{q}_t + \mathcal{Q}^{\boldsymbol{\theta}_M}(\mathbf{h}_{t+\Delta t}). \tag{3}$$

其中，Δt 是宏观传播器的时间步，Δt 是 δt的整数倍。宏观传播器通过时间反向传播 [51] 进行训练，以最小化组合均方误差（MSE）损失 $\|\tilde{}\mathbf{z}_{t+\Delta t} - \mathbf{z}_{t+\Delta t}\| + \|\tilde{}\mathbf{q}_{t+\Delta t} - \mathbf{q}_{t+\Delta t}\|$。可以为各个损失分量添加可选权重，以控制它们的相对重要性。

我们注意到，当前框架在演化潜在空间动力学的同时，也能预测关注物理量。此类关注量 $\tilde{}\mathbf{q}_{t+\Delta t}$ 可以通过还原到系统动力学的细尺度

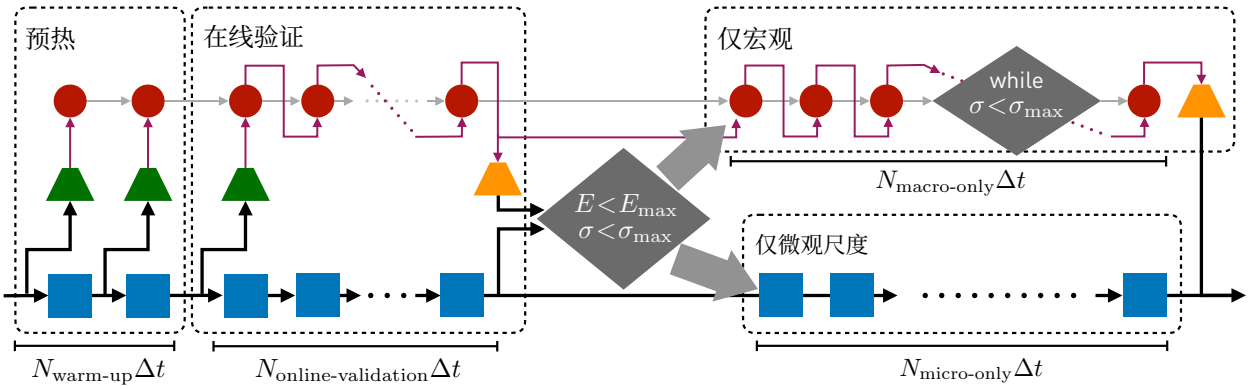

图 1：推理阶段，即 AdaLED 周期。 ■ 表示微观传播器， ● 宏观传播器、编码器、解码器，**黑色**线表示微观（高维）状态$\mathbf{x}_t$，▲紫色表示宏观（潜在）状态$\mathbf{z}_t$，以及灰色表示宏观传播器的隐状态$\mathbf{h}_t$。为简洁起见，关注量$\mathbf{q}_t$ 和外部强迫$\mathbf{f}_t$ 均未显示。根据在线验证阶段结束时模型的预测误差 E_t 和不确定性 σ_t，将执行仅宏观阶段或仅微观阶段。只要不确定性低于阈值（步数可能为 0），仅宏观阶段就会持续执行；也可以将其限制为给定的步数。

表征来计算，即 $\tilde{\mathbf{q}}_{t+\Delta t} = \mathcal{Q}(\mathcal{D}^{\boldsymbol{\theta}_\mathcal{D}}(\tilde{\mathbf{z}}_{t+\Delta t}))$. 然而，这种方法有两个缺点：（i）需要调用计算成本相对较高的解码器 $\mathcal{D}^{\boldsymbol{\theta}_\mathcal{D}}$；（ii）函数 $\mathcal{Q}$ 可能并未显式给出。因此，训练宏观传播器以直接预测 $\tilde{\mathbf{q}}_{t+\Delta t}$。此外，宏观传播器还会输出不确定性 $\sigma_{t+\Delta t} \in \mathbb{R}$，即

$$\sigma_{t+\Delta t} = \mathcal{S}^{\boldsymbol{\theta}_M}(\mathbf{h}_{t+\Delta t}). \tag{4}$$

不确定性用于稳健地控制微观传播器与宏观传播器之间的转换，具体解释见下文。

为了显著加速模拟，宏观传播器以时间步$\Delta t \gg \delta t$运行。在这里，编码器和解码器是卷积自编码器的两个部分，而宏观传播器则是由概率循环神经网络（PRNNs）组成的集成（见第 2.3 节）。值得注意的是，AdaLED 可以采用各种编码器和解码器，并兼容任何能够估计自身预测不确定性的宏观传播器。我们将编码器、解码器和传播器的组合称为机器学习模型（MLM）。

2.1. AdaLED 周期（推理）

AdaLED 的推理以迭代方式进行。在每次迭代中，AdaLED 都会评估 MLM 的准确度，并在准确度足够高时，暂时将模拟从微观尺度切换到宏观尺度。这种在不同尺度之间的交替，使微观传播器能够修正 MLM 引入的误差，并引导模拟回到流形 $\mathcal{M}_z$。此外，较短的循环使 MLM 能够替代轨迹中迄今已学会的部分模拟。对于训练数据中代表性不足或需要更长时间训练的其他部分，则由微观传播器处理。最后，频繁评估微观传播器也有助于持续收集训练数据。

AdaLED 中的每个计算周期包含三个阶段：(i) 预热阶段，(ii) 在线验证阶段，以及 (iii) 仅微观阶段或仅宏观阶段（图 1）。在预热 阶段，微观传播器和宏观传播器都会运行。在每个时间步，微观状态$\mathbf{x}_t$ 都会经过编码器 $\mathcal{E}^{\boldsymbol{\theta}_\mathcal{E}}$，并输入宏观传播器，以预热其隐状态$\mathbf{h}_t$（从$\mathbf{h}_t = \mathbf{0}$开始）。在在线验证 阶段，微观传播器和宏观传播器独立运行，以估计宏观传播器对系统轨迹当前片段的预测精度。在线验证阶段结束时，最终潜在状态 $\tilde{\mathbf{z}}_t$ 会被解码回高维空间，并计算一个特定于应用的 MLM 预测误差 $E_t = E(\mathbf{x}_t,\ \mathcal{D}^{\boldsymbol{\theta}_\mathcal{D}}(\tilde{\mathbf{z}}_t))$，用于衡量微观状态（真实值）与 MLM 预测之间的差异。如果 MLM 预测误差 E_t 或不确定性 σ_t 的

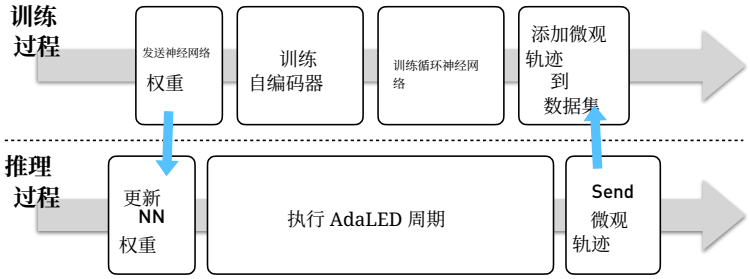

图 2：推理和训练循环。出于性能考虑，训练和推理可以选择在不同进程中执行。这种划分自然可以扩展到多进程训练、多进程模拟以及多个模拟。

宏观传播器超过转换阈值 $E_{\max}$ 和 $\sigma_{\max}$，则会舍弃宏观预测，模拟继续进入仅微观阶段。然而，如果二者均低于误差阈值，则接受宏观传播器的预测 $\tilde{\mathbf{z}}_t$，模拟继续进入仅宏观阶段。根据宏观预测是否被接受，将各个循环称为已接受 或已拒绝。

在已接受周期中，在线验证阶段之后是仅宏观 阶段，此时微观传播器暂停，唯一的计算是在潜在状态中使用低成本的宏观传播器完成的。只要预测不确定性 σ_t 低于阈值 $\sigma_{\max}$，该阶段就会持续。一旦超过阈值，该步的预测就会被丢弃。然后，将前一个时间步的潜在状态解码为高维状态，并传递给微观传播器。一个重要的假设是，微观传播器可以重新初始化为任意状态。此外，该阶段还可以选择限制为 $N^{\max}_{\text{macro-only}}$ 步。在图 1 中，数值 N_X 表示阶段 X中的时间步数量。

2.2. 数据集与训练

微观传播器生成的轨迹 $\mathbf{x}_t$ 被切分为长度为 L 步的轨迹，并存储在容量为 $D \gg 1$ 的动态数据集中。长度 L设为已接受周期中记录的状态数量： $L = 1 + N_{\text{warm-up}} + N_{\text{online-validation}}$。数据集填满后，添加新轨迹时，会随机均匀地选取一条现有轨迹并将其删除。随机删除轨迹可确保旧轨迹长期保留，从而缓解灾难性遗忘问题 [52], ，即神经网络在持续学习过程中对过去见过的样本的预测性能下降。

自编码器和宏观传播器在数据集的随机子集上分别依次进行训练。训练会持续进行，可以在每个 AdaLED 周期之后进行，也可以与 AdaLED 周期并行异步进行（图 2）。在推理过程中，一个 AdaLED 周期内，自编码器和宏观传播器的参数保持固定。

2.3. 预测不确定性估计

在 AdaLED 的宏观阶段，宏观传播器预测的轨迹最终会偏离微观传播器所产生的真实值，而偏离速率取决于系统动力学的复杂程度 [19]。在本研究中，我们不再手动选择宏步数，而是采用一种稳健机制来估计可靠粗粒化预测的持续时间。为此，我们使用概率网络和网络集成 [53]。

输入数据记为$\mathbf{x} \in \mathbb{R}^{N_x}$，网络的输出数据（目标值）记为$\mathbf{y} \in \mathbb{R}^{N_y}$，我们希望估计该网络的预测不确定性。系统的输出通过两个网络进行预测。第一个网络以 $\boldsymbol{\theta}_{\boldsymbol{\mu}}$ 为参数，输出均值 $\boldsymbol{\mu}^{\boldsymbol{\theta}_{\mu}}(\mathbf{x})$。该网络经过训练，以最小化目标值与均值输出之间的均方误差，即：

$$\ell_{\text{MSE}}(\boldsymbol{\theta}_{\boldsymbol{\mu}}, \mathbf{x}, \mathbf{y}) = \frac{1}{N_y} \sum_{i=1}^{N_y} \left(\boldsymbol{\mu}^{\boldsymbol{\theta}_{\mu}}(\mathbf{x})_i - y_i \right)^2 . \tag{5}$$

第二个网络以 $\boldsymbol{\theta_\sigma}$ 为参数，输出高斯分布的方差 $\boldsymbol{\Sigma}^{\boldsymbol{\theta_\sigma}}(\mathbf{x}) = \mathrm{diag}(\boldsymbol{\sigma}^{\boldsymbol{\theta_\sigma}}(\mathbf{x})^2)$，该分布的均值为 $\boldsymbol{\mu}^{\boldsymbol{\theta_\mu}}$，即 $p_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\mathbf{y};\ \boldsymbol{\mu}^{\boldsymbol{\theta_\mu}}(\mathbf{x})\ \boldsymbol{\Sigma}^{\boldsymbol{\theta_\sigma}}(\mathbf{x}))$ [54, 53]，其中 $\boldsymbol{\theta} = \{\boldsymbol{\theta_\mu}, \boldsymbol{\theta_\sigma}\}$。为简化起见，这里采用对角协方差矩阵。神经网络架构的详细信息见附录A。训练第二个网络时，通过最小化负对数似然损失（NLL）进行：

$$\begin{aligned}\ell_{\mathrm{NLL}}(\boldsymbol{\theta_\sigma}, \mathbf{x}, \mathbf{y}) &= -\log p_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{x}) \\ &= \frac{1}{2}\log\left(\boldsymbol{\sigma}^{\boldsymbol{\theta_\sigma}}(\mathbf{x})\right)^2 + \frac{\left(\mathbf{y} - \boldsymbol{\mu}^{\boldsymbol{\theta_\mu}}(\mathbf{x})\right)^2}{2\left(\boldsymbol{\sigma}^{\boldsymbol{\theta_\sigma}}(\mathbf{x})\right)^2} + \mathrm{const.}\end{aligned} \tag{6}$$

这些网络共同训练，可以视为一个参数为 $\boldsymbol{\theta}$ 的单一网络；当计算 NLL 损失时，权重 $\boldsymbol{\theta_\mu}$ 被视为固定值。总样本损失可写为：

$$\ell(\boldsymbol{\theta}, \mathbf{x}, \mathbf{y}) = \ell_{\mathrm{MSE}}(\boldsymbol{\theta_\mu}, \mathbf{x}, \mathbf{y}) + \ell_{\mathrm{NLL}}(\boldsymbol{\theta_\sigma}, \mathbf{x}, \mathbf{y}), \quad \boldsymbol{\theta} = \{\boldsymbol{\theta_\mu}, \boldsymbol{\theta_\sigma}\}. \tag{7}$$

这种将均方误差（MSE）与 NLL 损失结合、并对 $\boldsymbol{\theta_\mu}$ 使用解耦梯度的方法，比单独使用 NLL 损失取得了更高的准确度。

此外，我们考虑由 K 个此类概率网络组成的集成。每个网络均使用各自的参数 $\boldsymbol{\theta}^k$、$k \in \{1, \ldots, K\}$ 随机初始化，并在同一数据上分别训练，以最小化损失 $\ell(\boldsymbol{\theta}^k,\ \mathbf{x},\ \mathbf{y})$。对于给定的输入 $\mathbf{x}$，将输出 $\boldsymbol{\mu}^{(k)} = \boldsymbol{\mu}^{\boldsymbol{\theta}^k_\mu}(\mathbf{x})$ 和 $\boldsymbol{\sigma}^{(k)} = \boldsymbol{\sigma}^{\boldsymbol{\theta}^k_\sigma}(\mathbf{x})$ 组合，得到集成的最终预测 $\boldsymbol{\mu}(\mathbf{x})$ 和不确定性 $\boldsymbol{\sigma}(\mathbf{x})$，如下所示 [53]:

$$\begin{aligned}\boldsymbol{\mu}(\mathbf{x}) &= \frac{1}{K}\sum_k \boldsymbol{\mu}^{(k)}(\mathbf{x}), \\ \boldsymbol{\sigma}^2(\mathbf{x}) &= \underbrace{\frac{1}{K}\sum_k \left(\boldsymbol{\sigma}^{(k)}\right)^2(\mathbf{x})}_{\boldsymbol{\sigma}^2_{\mathrm{ind}}(\mathbf{x})} + \underbrace{\frac{1}{K}\sum_k \left(\boldsymbol{\mu}^{(k)}\right)^2(\mathbf{x}) - \boldsymbol{\mu}^2(\mathbf{x})}_{\boldsymbol{\sigma}^2_{\mathrm{std}}(\mathbf{x})}.\end{aligned} \tag{8}$$

式（8）中的项 $\boldsymbol{\sigma}_{\mathrm{ind}}$ 指的是各个网络分别估计的预测不确定性。另一方面，项 $\boldsymbol{\sigma}_{\mathrm{std}}$ 衡量集成在预测 $\mathbf{y}$ 时的分歧。因此，该项为给定输入 $\mathbf{x}$ 的训练误差提供了估计。对于未见过的状态 $\mathbf{x}$ 或在训练数据中代表性不足的状态，我们预计 $\boldsymbol{\sigma}_{\mathrm{ind}}$ 和 $\boldsymbol{\sigma}_{\mathrm{std}}$ 会大于频繁出现的状态所对应的值。由向量不确定性 $\boldsymbol{\sigma}$ 可定义标量不确定性 σ，其定义方式为 $\sigma = \|\boldsymbol{\sigma}\|_2$ 或 $\sigma = \|\boldsymbol{\sigma}\|_2 / N_y$。

2.4. AdaLED 超参数

误差阈值 $E_{\max}$ 和 $\sigma_{\max}$ 因应用而异，并决定了加速与准确度之间的权衡。自编码器的潜在状态维度 d_z 应根据系统动力学及其有效自由度 [24]来选择。其余超参数（如网络规模和层数）可通过小规模实验和超参数调优来确定。

3. 案例研究：范德波尔振子

我们首先展示 AdaLED 在范德波尔振子（VdP）上的能力。该系统被用作多种多尺度框架的基准 [15, 10, 13]。与这些框架不同，我们不预先区分快速动力学和慢速动力学。相反，我们任意改变振子的极限环和振荡时间尺度；后者由阻尼参数 μ控制，以此展示 AdaLED 能够适应这些变化。在此案例研究中，我们不使用自编码器，即编码器和解码器均为恒等算子。

范德波尔振子 [55, 56] 是一种由以下方程描述的非线性阻尼振荡器：

$$\begin{aligned}\frac{\mathrm{d}x}{\mathrm{d}t} &= \mu\left(x - \frac{1}{3}x^3 - y\right), \\ \frac{\mathrm{d}y}{\mathrm{d}t} &= \frac{1}{\mu}x,\end{aligned} \tag{9}$$

其中，$\mu = \mu(t) > 0$ 是一个随时间变化的系统参数，用于控制系统的非线性和阻尼。

微观传播器是一种基于欧拉方法的 ODE 积分器，从随机初始条件（x_0，y_0）$\sim \mathcal{U}([-5,5]^2)$开始，以 $\delta t = 0.001$ 的时间步积分式 (9)。宏观传播器由五个长短期记忆网络组成。其架构详见附录 A。该集成经过训练，可用宏观时间步长 $\Delta t = 0.1$预测动力学。每个长短期记忆网络将元组 $(x(t)\ y(t)\ \mu(t))$ 作为输入，并输出残差 $\Delta x(t) = x(t+\Delta t) - x(t)$ 和$\Delta y(t) = y(t+\Delta t) - y(t)$ 的均值和方差 $(\mu_{\Delta x}(t)\ \mu_{\Delta y}(t)\ \sigma^2_{\Delta x}(t)\ \sigma^2_{\Delta y}(t))$。

对于恒定的 μ，系统会进入一个形状取决于 μ 的极限环（图 B.1）。对于此处考虑的μ 和 Δt 值，完成一个极限环大约需要 60 到 90 个宏观时间步 Δt。

AdaLED 周期配置为包含 7 个预热步和 25 个在线验证步。为避免 AdaLED 周期与系统的极限环同步，每个 AdaLED 周期的宏观步数和微步数上限均从 80 到 120 步之间均匀随机选取。我们观察到，这种同步会导致训练数据集被同一极限环区域的数据填满，而循环的其余部分则代表性不足。对于给定的循环 c，我们将 宏观利用率 $\eta_c = N^c_{\text{macro-only}}/N^c$定义为仅宏观阶段所执行步数的比例，其中 $N^c_{\text{macro-only}}$ 为仅宏观步数，N^c 为循环的总步数 c。在被拒绝的循环中，$\eta_c = 0$。总宏观利用率η 定义为 $\eta = (\sum_c N^c_{\text{macro-only}})/(\sum_c N^c)$。根据所选的阶段时长，总宏观利用率的最大可达值为 $\eta \approx 75.6\%$。

数据集的最大容量设为 1280 条轨迹。训练以固定速率进行，即每个模拟宏步训练 2 条轨迹。

预测误差定义为 $E = \sqrt{(x_{\text{micro}} - x_{\text{macro}})^2 + (y_{\text{micro}} - y_{\text{macro}})^2}$，预测不确定性定义为 $\sigma = \sqrt{\sigma^2_{\Delta x} + \sigma^2_{\Delta y}}$。AdaLED 的转换阈值设为 $E_{\max} = 0.10$ 和 $\sigma_{\max} = 0.10$。需要注意的是，误差阈值 $E_{\max}$ 指的是 25 个在线验证步之后累积的误差，而不是单步预测误差。

3.1. 结果

我们分析 AdaLED 在以下三种不同的 $\mu(t)$ 情况下的性能：

- $\mu_{\text{ALT}}(t)$，一种在 $\mu = 1.0$ 和 $\mu = 3.0$ 两个值之间交替、每 5000 个时间步切换一次的分段常数函数；
- $\mu_{\text{RAND}}(t)$，一种取多个随机选定值 $\mu \in [1.0, 3.0]$的分段常数函数；以及
- $\mu_{\text{BROWN}}(t)$，一种添加了类布朗噪声的分段常数函数 $\mu_{\text{RAND}}(t)$（详情 ex

-

（详见附录 B）。

系统经过 400000 个时间步进行积分。长短期记忆网络的超参数通过一项初步研究进行调整，相关研究结果见附录 B.1。

三个案例的宏观利用率以及函数 $\mu(t)$ 均如图 3 所示。案例$\mu_{\text{ALT}}(t)$ 显示在顶部图中。运行开始时，宏观利用率为零，这符合预期，因为宏观传播器尚未训练，无法给出准确的预测。约 8000 个时间步后，预测误差达到预期阈值，AdaLED 开始接受宏观传播器的预测。因此，宏观利用率逐渐升至 60%。在第 50 000 个时间步，$\mu(t)$ 的值突然改变，使系统进入一个未见状态。AdaLED 正确检测到这一变化，并开始拒绝宏观传播器的预测，因为这些预测突然变得不可靠。然而，不久之后，宏观传播器便学会了新的状态区间，其预测再次被接受，宏观利用率随之上升，并接近最大值 75。6%。又经过 50 000 个时间步后，系统切换回 $\mu = 3$。宏观传播器此前已经观测并学会了这一状态区间，正如 AdaLED 恢复运行后接受率一直很高、运行未曾中断这一事实所表明的那样。

图 3 中间图所示的情形 $\mu_{\text{RAND}}(t)$ 的结果与前一种情形表现出相似的行为。在模拟开始时，所有循环均被拒绝，直到网络学会

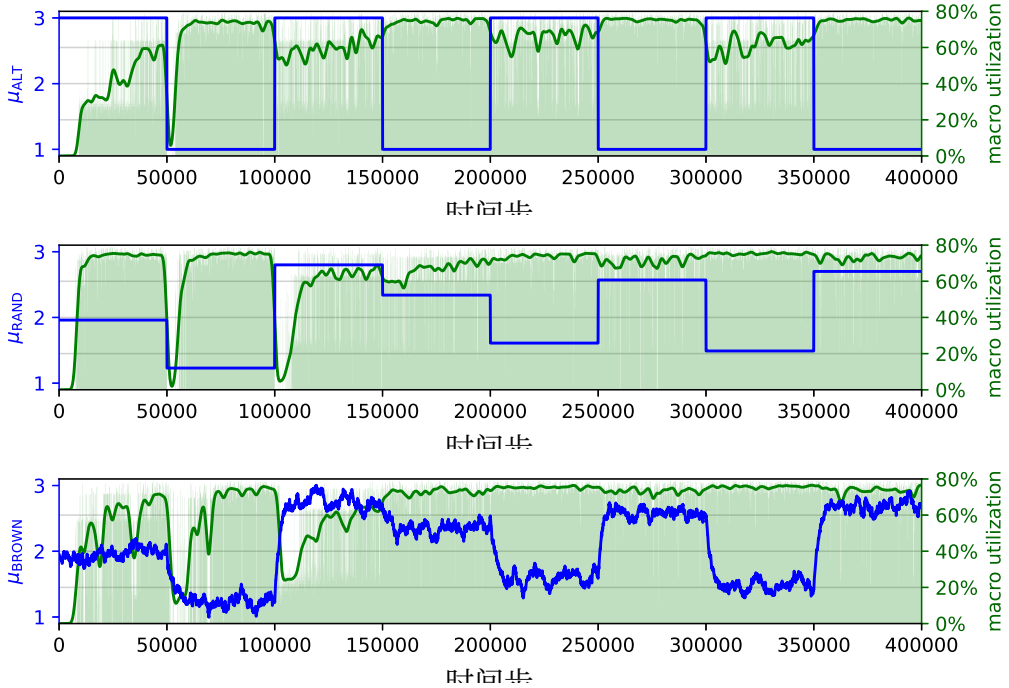

图 3：范德波尔振子案例研究中，三种不同的 $\mu(t)$ 变体的宏观利用率（即在仅宏观阶段执行的步数占总步数的比例）。浅绿色直方图显示每个独立 AdaLED 周期的宏观利用率，深绿色曲线则显示经过高斯模糊平滑处理后的宏观利用率（仅用于可视化）。

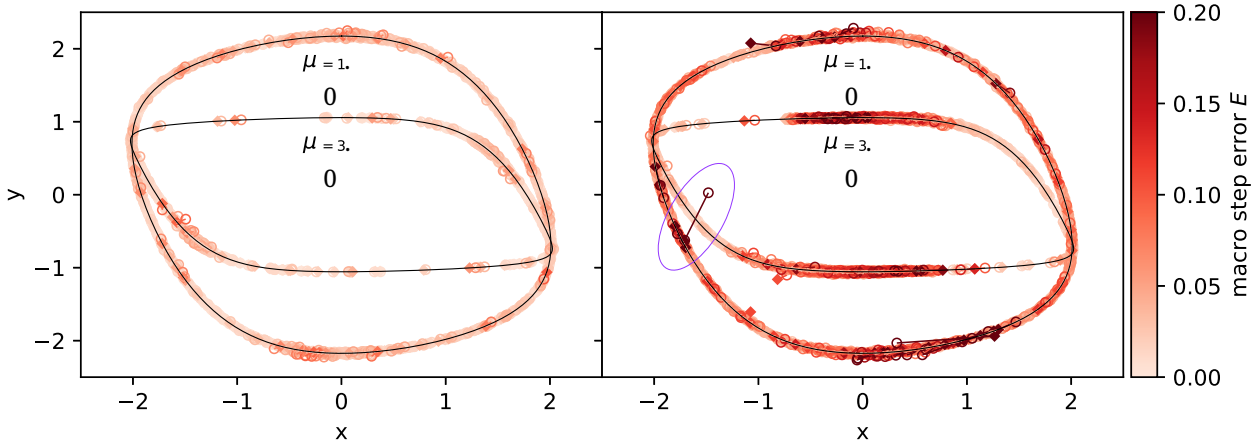

图 4：范德波尔振子案例研究中案例 $\mu_{\mathrm{ALT}}(t)$ 每个已接受周期的预测误差可视化。左：在线验证阶段结束时的误差；右：仅宏观阶段结束时的误差。菱形表示微观状态，空心圆表示宏观状态，颜色表示误差 E（欧几里得距离）。端点位于极限环之外的 AdaLED 周期，对应于 $\mu_{\mathrm{ALT}}(t)$ 的值发生变化的周期。时间步 50 000 处的首次此类变化（圈出的点对）在图 5 中进行了详细展示。

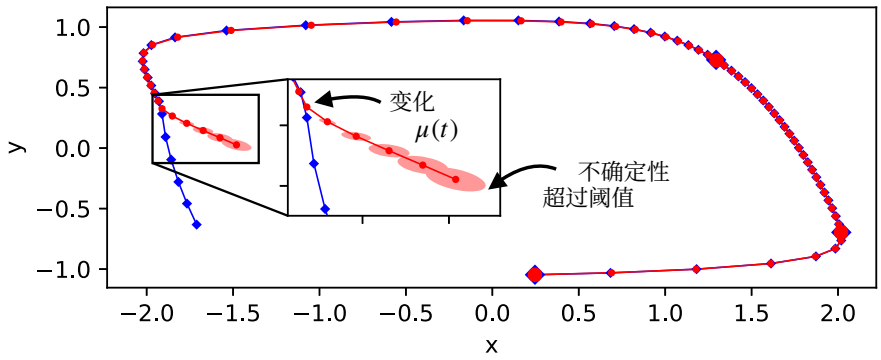

图 5：在时间步 50000 附近的 AdaLED 周期，此时 $\mu_{\mathrm{ALT}}(t)$ 从数值 3.0 变为 1.0。蓝色菱形 表示微观状态，红色圆圈 ● 表示宏观状态，红色椭圆表示集成协方差。图中显示，由于集成中的各个长短期记忆网络对数值 $\mu(t)$的响应各不相同，AdaLED 很快检测到了动力学变化，并停止了宏观传播器的运行。

相应的状态区间。在 μ 首次发生两次变化后，这一趋势再次出现。从第三次变化开始，我们观察到 AdaLED 能够在先前见过的 μ 数值之间插值，并且尽管 μ_{RAND}（t）发生变化，仍能继续给出准确的预测。这表明 AdaLED 能够自适应地学习并插值至未见过的动力学状态。

最后，图 3 底部的图展示了更复杂的含噪 μ_{BROWN}（t）情况。AdaLED 再次逐步学会取代微观传播器，并自适应地学习不同的动力学状态。然而，与 μ_{ALT} 和 μ_{RAND} 的情况相比，宏观传播器达到最高接受率所需的时间更长。我们认为这是因为学习任务更难：数据由多个时间尺度各异的极限环生成。

图 4 展示了案例 $\mu_{\mathrm{ALT}}(t)$ 中每个已接受周期的在线验证误差和最终测试误差（仅宏观阶段结束时的预测误差）。测试误差的计算方式是：即使在仅宏观阶段，也运行微观传播器，但仅用于评估。在生产运行中，仅宏观阶段期间微观传播器处于非活动状态。

我们观察到，误差总体较小，几乎每个循环结束时都非常接近极限环。 $\mu(t)$ 突然变化时则会出现例外。值得注意的是，在线验证误差（左图）可通过调整 $E_{\max}$ 的值直接控制。另一方面，测试误差（右图）只能通过 $E_{\max}$ 和 $\sigma_{\max}$ 间接控制，因此可用作衡量代理模型稳健性和质量的指标。这凸显了 AdaLED 相较于其他多尺度框架的优势，因为它能够预先控制误差阈值。

系统在 μ_{ALT}首次发生变化期间的行为（变化发生于第 50 000 个时间步）如图 5 所示。在此变化之前，长短期记忆网络仅使用具有 $\mu = 3$的轨迹进行训练。因此，对于这一特定状态区间，集成中各个网络的预测彼此相似且准确。然而，一旦 $\mu_{\mathrm{ALT}}(t)$的值发生变化，集成中的不同长短期记忆网络对 $\mu = 1$的响应便各不相同，因为对于未见状态的预测是任意的，并且取决于权重初始化。这会导致预测逐渐分歧，不确定性随之增加（定义见式 (8)）。七步之后，当不确定性超过阈值 $\sigma_{\max}$ 时，该循环便终止。第 7 步被舍弃，因此未显示在图中。

关于测试误差对不确定性阈值 $\sigma_{\max}$ 和集成规模 K 的依赖性的更多结果见附录 B.2。

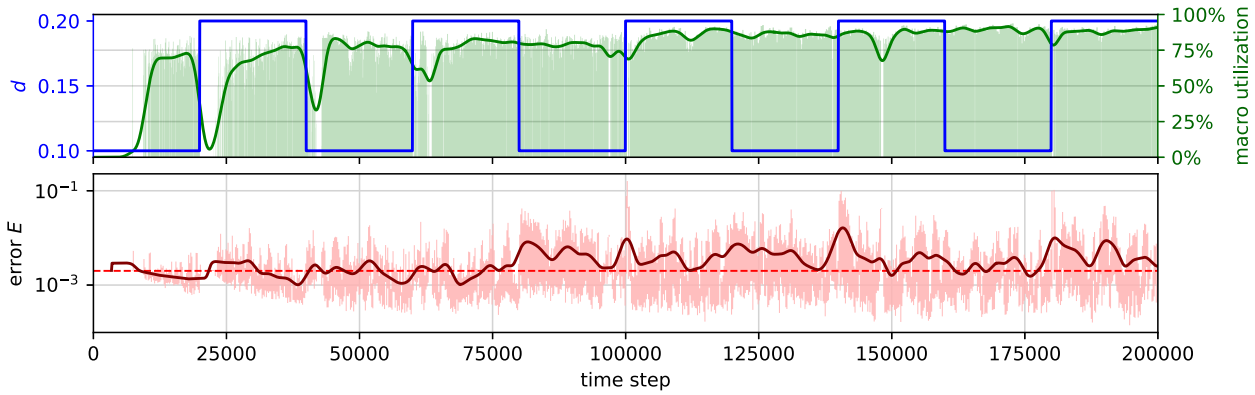

图 6：AdaLED 在扩散参数随时间变化的反应-扩散案例研究中的性能 d （蓝线）。上图：宏观利用率（在仅宏观阶段执行的步数占比）。浅绿色直方图显示每个 AdaLED 周期的宏观利用率，深绿色曲线显示经高斯模糊平滑后的宏观利用率（仅用于可视化）。下图：重构状态 $\mathbf{w}$ 的测试误差。每步误差（淡红色）在仅宏观阶段开始时的低值与该阶段结束时的较高误差之间交替变化。深红色表示平滑后的测试误差，红色虚线表示在线验证阈值 $E_{\max}$。

4. 案例研究：反应扩散方程

在此，我们在由以下方程描述的 lambda-omega 反应-扩散系统 [57, 58] 上测试 AdaLED ：

$$\begin{aligned}\frac{\partial u}{\partial t} &= [1-(u^2+v^2)]u + \beta(u^2+v^2)v + d_1\nabla^2 u,\\ \frac{\partial v}{\partial t} &= -\beta(u^2+v^2)u + [1-(u^2+v^2)]v + d_2\nabla^2 v\end{aligned} \tag{10}$$

对于 $-10 \le x,\ y \le 10$，其中 $\beta = 1.0$ 是反应参数， $d_1 = d_2 = d = d(t)$ 是随时间变化的扩散参数。该方程在 96×96 均匀网格上采用四阶 Runge–Kutta–Fehlberg 方法进行积分，时间步为 $\Delta t = 0.05$。因此，系统的状态由张量 $\mathbf{w} = (\mathbf{u},\ \mathbf{v}) \in \mathbb{R}^{2\times 96\times 96}$ 完整描述。该系统呈现出螺旋波，其形状取决于参数 d。

我们评估了扩散参数 d 每隔 20 000 个时间步在 0.1 和 0.2 之间交替变化时 AdaLED 的性能。AdaLED 周期配置为包含 5 个预热步、18 个在线验证步、10 至 15 个仅微观步数，以及 400 至 500 个仅宏观步数。AdaLED 的转换阈值设为$E_{\max} = 0.002$ 和 $\sigma_{\max} = 0.002$。我们使用宏观状态与微观状态 $\mathbf{w}$ 之间的均方误差（MSE）作为误差指标，即 $E = E(\mathbf{w}\ \tilde{\mathbf{w}}) = \|\mathbf{w} - \tilde{\mathbf{w}}\|_2^2/(2\cdot 96\cdot 96)$。模拟运行 200 000 个时间步。系统、神经网络和训练的其他细节列于附录 C。

宏观利用率 η 和测试误差 E 如图 6 所示。与范德波尔振子案例研究类似，我们观察到，起初宏观利用率为零，因为网络尚未训练完成。然而，约经过 10 000 个时间步后，自编码器学会了重构状态，长短期记忆网络也学会了预测动力学。因此，AdaLED 开始接受宏观预测。当扩散参数 d 从 0.1 变为 0.2 时，AdaLED 识别出宏观预测不可靠，并切换回模拟器。学习到新的状态区间后，AdaLED 恢复使用宏观传播器。图 6 底部的图显示了测试误差 E。总体而言，宏观利用率达到 75% $\pm$1%，MSE 为 0.0055 ± 0.0013（相对均方误差为 0.012 ± 0.003）。报告的置信水平基于十次重复模拟运行计算出的方差。

图7展示了模拟在第 200 000 个时间步时的快照，此时仅宏观阶段已进行了 305 个时间步。预测状态与预期状态一致，这表明即使在潜在空间中进行了大量时间步的模拟，宏观推进器仍具有较高的准确度。为进行测试，在仅宏观阶段期间继续运行微观模拟，以获取该阶段的微观状态。

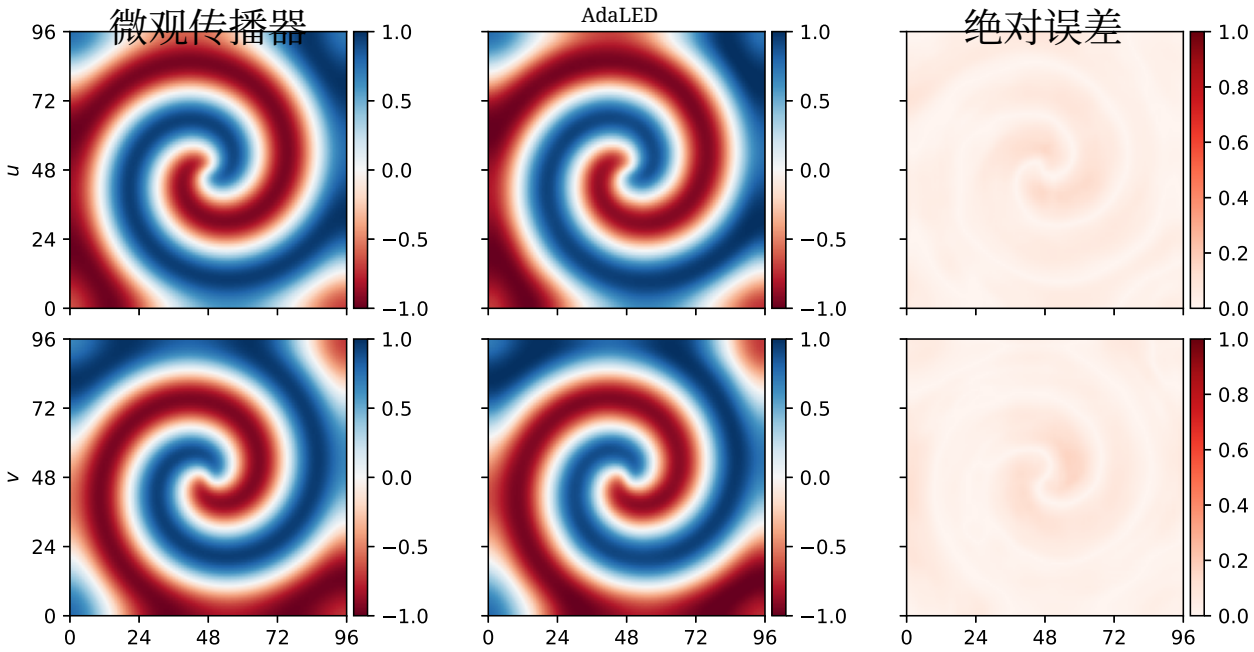

图7：时间步 200000 时的模拟快照，此时进入仅宏观阶段已 305 步。左：微观传播器（真实值），中：AdaLED 机器学习模型，右：绝对误差。均方误差为 $E = 0.0021$ （相对误差为 0.0045）。

5. 案例研究：二维圆柱绕流

最后，我们使用 AdaLED 加速对不同雷诺数下绕圆柱体流动的二维直接数值模拟（DNS）。AdaLED 经过训练，可预测代表流动状态的速度场。预测完整的模拟状态，使宏观（潜在）尺度与微观尺度（DNS）之间能够交替进行。除了预测状态，AdaLED 还需要预测流体作用于圆柱体上的力。该力是我们希望在模拟的所有时间步都能获取的关注量 $\mathbf{q}(t)$。

该系统由不可压缩纳维-斯托克斯方程支配，并通过 Brinkman 惩罚法施加无滑移边界条件 [59]:

$$\nabla \cdot \mathbf{u} = 0, \tag{11}$$

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} = -\frac{1}{\rho}\nabla p + \nu \nabla^2 \mathbf{u} + \lambda(\mathbf{u}^s - \mathbf{u})\chi, \tag{12}$$

其中，$\mathbf{u} = \mathbf{u}(\mathbf{x}\ t)$为流体速度场，$\rho = 1$ 为流体密度，$p = p(\mathbf{x}\ t)$为压力，$\nu = 10^{-4}$为运动黏度，$\lambda = 10^6$ 为布林克曼惩罚系数，$\mathbf{u}^s = \mathbf{u}^s(t)$为圆柱体的速度，$\chi = \chi(\mathbf{x}\ t)$为圆柱体的特征函数，在圆柱体内部等于 1，在外部等于 0。该方程在具有开放边界条件的 $[0,1] \times [0,0.5]$ 域上求解。直径为 $d = 0.075$ 的实心圆柱体固定在相对于模拟域的坐标 (0.2,0.25) 处。圆柱体和模拟域相对于流体以 $u_x^s(t) = \mathrm{Re}(t)\nu/d$ 的速度水平移动，其中雷诺数 $\mathrm{Re}(t)$（外部强迫）在 $\mathrm{Re} = 400$ 和 $\mathrm{Re} = 1200$之间变化。在此范围内，对于固定的 Re，圆柱体后方会形成涡街。

公式（11）和（12）采用压力投影方法 [60]在最大分辨率为 1024×512 个网格单元的自适应笛卡尔网格上求解。对于本研究，自适应非均匀网格被插值至 1024×512 个网格单元的最大分辨率，因此在 AdaLED 使用时将其视为均匀网格。

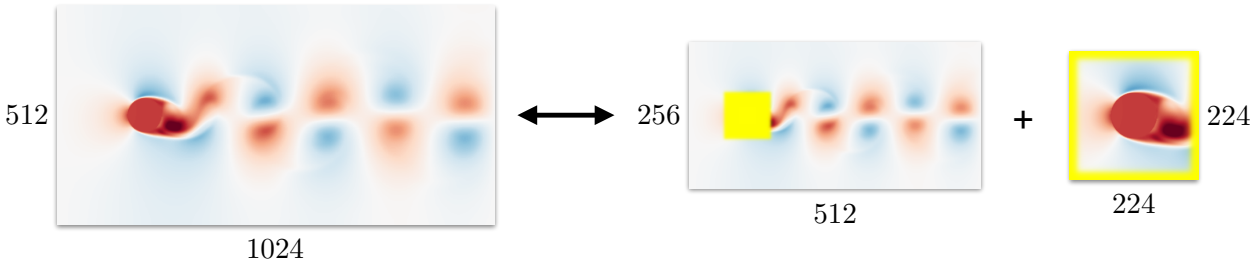

图 8：多分辨率 AE 示意图。为加快训练并降低存储和内存需求，AE 对速度场的两个降采样变体进行处理。黄色区域表示用于重建完整分辨率场的混合掩码。

为了降低高维网格的计算需求并加快 AdaLED 的训练过程，我们提出了一种新颖的基于物理学的多分辨率自编码器（AE）。所提出的 AE 利用了流动的特性，在远离圆柱体的区域采用较低分辨率的网格；与圆柱体附近相比，该区域的流动呈现出更简单的特征。具体而言，AE 处理两个降采样网格：一个覆盖整个区域的半分辨率网格，以及一个位于圆柱体周围的小型全分辨率补丁（图 8）。通过采用这种多分辨率方法，我们能够降低存储和内存需求，并加快训练速度。此外，AE 输出的是流函数，而不是速度场 [61]。这种受物理学启发的架构确保速度场的散度为零（根据公式（11））。我们还加入了额外的基于物理学的涡量损失，以提升性能。具体细节见附录 D.1。

除了用于重建系统动力学并支持宏观到微观转换所必需的潜在状态 $\mathbf{z}(t)$ 外，MLM 还会输出流体作用于圆柱体上的力$\mathbf{F}^{\text{cyl}}(t)$，将其作为关注量 $\mathbf{q}(t)$。在宏观传播器的训练损失和不确定性估计中，$\mathbf{F}^{\text{cyl}}$ 与$\mathbf{z}$ 的相对重要性可通过用某个因子 α_F 对力进行缩放来控制。

宏观传播器由五个概率长短期记忆网络组成。每个长短期记忆网络都经过训练，用于预测残差的均值和方差 $\Delta\mathbf{z}(t) = \mathbf{z}(t+\Delta t) - \mathbf{z}(t)$ 以及 $\Delta\mathbf{q}(t) = \mathbf{q}(t+\Delta t) - \mathbf{q}(t)$ 、 $\mathbf{q}(t) = \alpha_F \mathbf{F}^{\text{cyl}}(t)$。验收标准基于速度场的相对重建误差 E

$$E(\tilde{\mathbf{u}}, \mathbf{u}) = \frac{\|\mathbf{u} - \tilde{\mathbf{u}}\|_2^2}{\|\mathbf{u}\|_2^2}, \quad \|\mathbf{u}\|_2^2 = \sum_{ij} \mathbf{u}_{ij}^2, \tag{13}$$

其中，$\mathbf{u}$ 是微观传播器输出的全分辨率速度场，而 $\tilde{\mathbf{u}} = \mathcal{D}^{\theta_{\mathcal{D}}}(\tilde{\mathbf{z}})$是机器学习模型的预测结果。自编码器本身采用另一种损失函数进行训练，详见附录 D.1。 σ 宏观传播器预测结果的总不确定性定义为不确定性向量$\sigma = \sqrt{((\boldsymbol{\sigma}_{\Delta\mathbf{z}}(t))^2 + (\boldsymbol{\sigma}_{\Delta\mathbf{q}})(t)^2) / (d_z + 2)}$的标准差。

AdaLED 的时间步设置为 $\Delta t = 0.005$，因此对于雷诺数 $\text{Re} = 1000$，每个涡街周期约包含 60 个 AdaLED 时间步。为简单起见，微观传播器的内部时间步 δt 在整个模拟过程中保持固定。对于 $\text{Re}(t)$ 不超过 1000 的模拟，$\delta t = 0.005/18$；对于 $\text{Re}(t)$ 不超过 1200 的模拟，$\delta t = 0.005/21$，从而得到 ~ 0.4的库朗数。

我们采用由 4 个预热步、12 个在线验证步、400 至 500 个宏步以及 9 至 14 个微步组成的 AdaLED 周期。为避免 AdaLED 周期与涡街周期同步，每个周期的两个边界值均以均匀分布随机选取。数据集容量设置为 256 条轨迹。为最大限度提高 AdaLED 的加速比，训练在独立的计算节点上进行，并与推理和微观传播器并行运行，如图 2 所示。实验在 Piz Daint 超级计算机上进行，使用两个 XC50 节点；每个节点配备一块运行频率为 2.6 GHz 的 12 核 Intel Xeon E5-2690 CPU 和一块 16GB Nvidia P100 GPU。模拟使用 CubismAMR 软件 [62] 执行。

5.1. 结果

在第 5.1.1 节中，我们展示了 AdaLED 在不牺牲准确度的情况下加速圆柱绕流模拟的有效性。在第 5.1.2 节中，我们强调了自适应训练对于具有变化的动力学的系统的重要性。最后，在第 5.1.3 节中，我们开展消融研究，以评估多分辨率自编码器的优势。

5.1.1. AdaLED 的有效性

我们总共进行了 300 000 个时间步的模拟，其中雷诺数每 5000 个时间步在 $\mathrm{Re} = 600$、$\mathrm{Re} = 750$ 和 $\mathrm{Re} = 900$ 之间循环变化。表 2 列出了超参数，这些参数根据较短模拟的性能进行了调节，相关结果见第 5.1.3 节。对于此次模拟，误差阈值和不确定性阈值分别设为 $E_{\max} = 0.017$ 和 $\sigma^2_{\max} = 0.00035$。需要指出的是，这些是 AdaLED 的关键超参数，可根据需要在准确度和加速效果之间进行权衡。

图 9 显示了 $\mathrm{Re}(t)$ 剖面、宏观利用率和误差。误差分别对应在线验证阶段的验证误差以及仅宏观阶段的测试误差。为计算这些误差，我们将微观状态 $\mathbf{u}_t$ 与 MLM 生成的重构状态 $\tilde{\mathbf{u}}_t = \mathcal{D}^{\theta_\mathcal{D}}(\tilde{\mathbf{z}}_t)$ 进行比较。我们使用公式 (13) 中的速度场误差度量 E 来量化速度场误差。对于力 $\mathbf{F}_{\mathrm{cyl}}$ 误差，我们将归一化误差 E_F 定义如下：

$$E_F = E_F(\mathbf{F}'_{\mathrm{cyl}}, \mathbf{F}_{\mathrm{cyl}}) = \frac{\left\|\mathbf{F}'_{\mathrm{cyl}} - \mathbf{F}_{\mathrm{cyl}}\right\|}{F^{\mathrm{avg}}_{\mathrm{cyl}}}, \tag{14}$$

其中，$F^{\mathrm{avg}}_{\text{圆柱}} = \langle \|\mathbf{F}_{\mathrm{cyl}}\| \rangle \approx 0.079$ 是力的平均幅值。出于测试目的，为了获取微观状态 $\mathbf{u}_t$，即使在仅宏观阶段，我们仍继续运行模拟。在此期间，MLM 的训练会暂时暂停。

在图 9 中，我们观察到与前两个案例研究相同的趋势。初始时，宏观利用率为零。网络训练充分后，框架开始接受 MLM 的预测。随着训练继续进行，误差和不确定性降低，宏观利用率随之提高。在仅宏观阶段，测试误差 E 和 E_F 保持较低水平，平均分别为 1% 和 5%。以降低加速比为代价，还可以进一步降低误差。

在图 10 中，我们更仔细地考察了误差和不确定性在轨迹选定区段内的变化情况，具体而言，是从雷诺数 $= 900$ 到雷诺数 $= 600$的过渡过程。宏观预测是否被接受取决于误差 E 及其阈值 $E_{\max}$。在雷诺数过渡之前，在线验证误差始终低于阈值，因此 AdaLED 接受宏观预测。然而，在持续数百个时间步的过渡期内，MLM 无法可靠地预测动力学。因此，AdaLED 转而使用微观传播器。过渡期结束后，AdaLED 恢复使用宏观传播器。需要注意的是，误差 E 有时可能会超过阈值 $E_{\max}$。阈值 $E_{\max}$ 表示在线验证阶段结束时的最大误差，因此应将其设置为低于期望的最大可容许误差的值。

与控制 AdaLED 是否进入仅宏观阶段的误差 E 相比，不确定性 σ 和阈值 $\sigma_{\max}$ 控制着该阶段的持续时间。一旦 σ 超过 $\sigma_{\max}$，仅宏观阶段就会停止。

预测力 $\mathbf{F}_{\mathrm{cyl}}$ 的准确度如图 10 底部的图所示。我们观察到两条力曲线之间具有良好的一致性。潜在轨迹的可视化结果见附录 D.5 中的图D.4。

图 11 展示了仅宏观阶段期间模拟的一个快照。我们观察到，AdaLED 准确地复现了模拟的状态，并以很高的准确度捕捉到了流动的特征。值得注意的是，误差集中在流动的细尺度结构上。可以认为，误差分布中的双弧表明，该误差可能部分归因于宏观传播器以不正确的速度推进动力学。

图 12 对比了独立模拟与 AdaLED 加速模拟的执行时间。绿色区域表示使用 AdaLED 节省的时间。整个运行过程（300 000 个时间步）的总宏观利用率为 69%，加速比约为 2.9 倍。训练

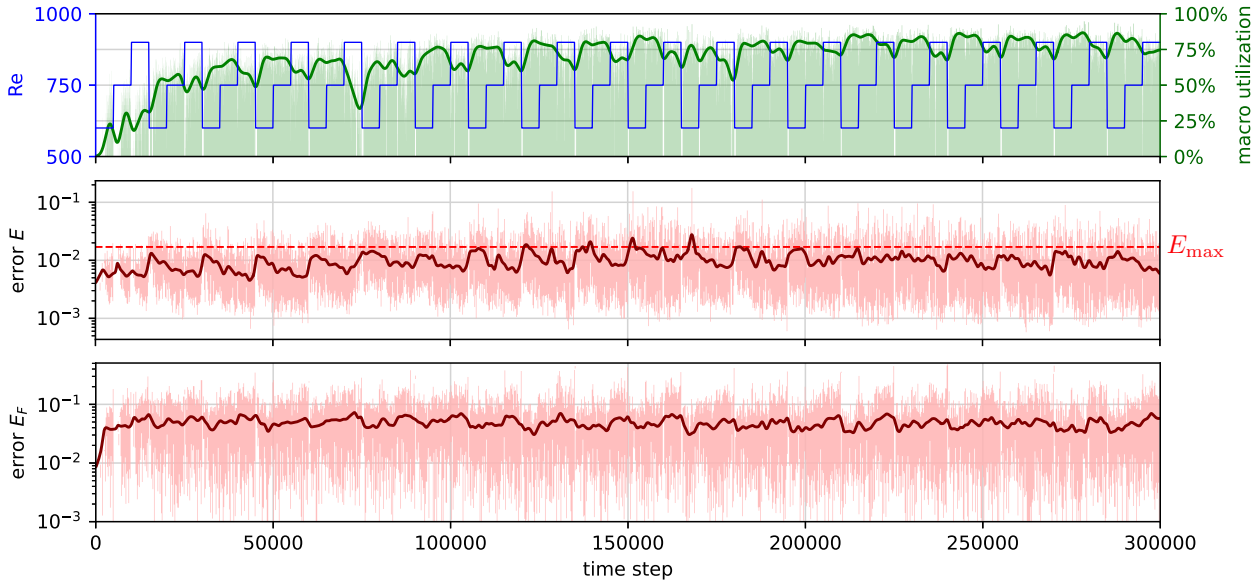

图 9：AdaLED 在 $\mathrm{Re}(t) \in \{600, 750, 900\}$ 的圆柱体后方流动模拟中的性能（第 5.1.1 节）。顶部：雷诺数 $\mathrm{Re}(t)$ 剖面和宏观利用率 η。中部和底部：速度的验证误差（E，公式 (13)）以及圆柱体上的力的验证误差（E_F，公式 (14)）。每步误差（淡红色）在仅宏观阶段开始时的低值和仅宏观阶段结束时的较高误差之间交替变化。速度误差平均值接近 1%（深红色），力误差接近 5%（互相关为 0.99）。这些误差仅指仅宏观步数。图 10 展示了短模拟片段中误差的详细视图。

收敛后，最后 15 000 个时间步的宏观利用率达到 80%，加速比为 4.3 倍。这表明，经过训练的 MLM 可应用于其他具有 $\mathrm{Re} \in \{600, 750, 900\}$ 的模拟，并取得类似的性能。

表1展示了每个模拟时间步的执行时间分解。对于给定的雷诺数(t)剖面，单独运行微观传播器时，每个时间步平均耗时969毫秒（雷诺数越大，模拟速度越慢）。在 AdaLED 中使用时，首次发生微观到宏观转换后，由于自编码器重构不够完善，网格尺寸略有增加，因此微观传播器的速度降至每步1055毫秒（+9%）。AdaLED 的平均开销（自编码器、日志记录、诊断等）相对较小，平均每个时间步耗时42毫秒。最后，宏观传播器每个时间步仅需5毫秒。

我们得出结论，对于计算成本高昂的计算流体力学模拟，部署 AdaLED 的开销极小。在这种情况下，加速比实际上仅由宏观利用率决定。提高误差（增大误差阈值）或采用准确度更高的模型，都能实现更高的加速比。后者可通过改进网络架构、采用更有效的训练过程、扩大集成规模或进行更全面的调优来实现。

不同雷诺数剖面下的一次运行结果见附录D.6。

5.1.2. 自适应性的重要性

下文将展示自适应性的重要性，即在整个模拟过程中持续训练，并适应新的状态和轨迹；我们将其与预训练或仅训练至某一时间点进行比较。我们分析了两种时变雷诺数剖面 $\mathrm{Re}(t)$。第一种情况下，$\mathrm{Re}(t)$ 在整个模拟过程中以锯齿状方式在 500、750 和 1000 之间切换。第二种情况下，$\mathrm{Re}(t)$ 的初始变化与第一种剖面相同，但在模拟的后半段切换到不同的状态区间（400、600、800、1000 和 1200），以模拟系统在模拟后期进入新状态区间的情形。对于每种剖面，我们测试了两种设置：一种是在整个过程中启用训练（自适应），另一种仅在模拟的前半段启用训练（非自适应）。

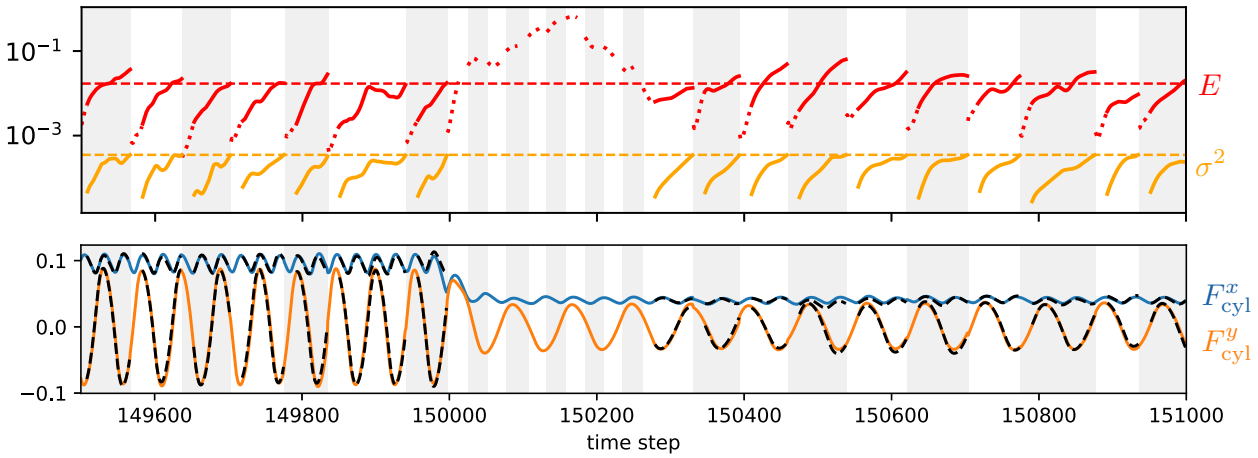

图 10：第 5.1.1 节和图 9 中圆柱后方流动模拟的一部分的详细视图，展示雷诺数 $= 900$ 到雷诺数 $= 600$ 的过渡过程。上图：速度验证误差 E 和不确定性的平方 σ^2（预热阶段和在线验证阶段用点线表示，仅宏观阶段用实线表示），以及它们的阈值 $E_{\max}$ 和 $\sigma^2_{\max}$（虚线）。下图：圆柱体上的水平力（蓝色）、垂直力（橙色）和宏观模型的预测（黑色虚线）。

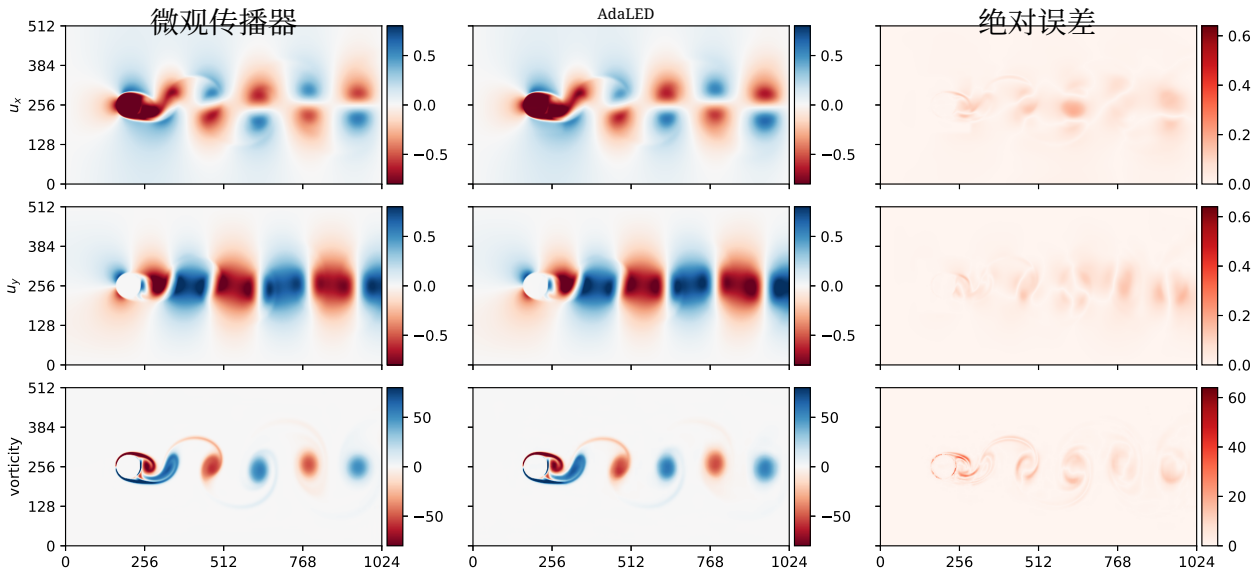

图 11：第 5.1.1 节模拟在时间步 212500（雷诺数 $= 600$）时的快照，此时已进入仅宏观阶段 50 个时间步，相对误差为 $E \approx 0.014$（公式 (13)）。左：微观传播器状态 $\mathbf{u}_t$ 和涡量 ω_t；中：代理模型（MLM）的预测和全分辨率重建；右：绝对误差。

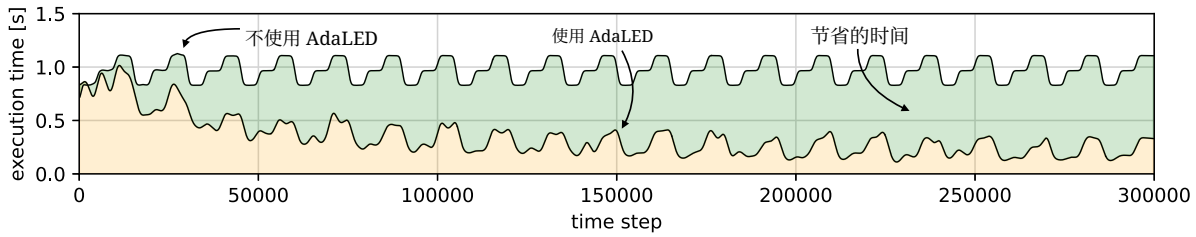

图12：第 5.1.1 节仿真设置下，使用和未使用 AdaLED 时平滑后的时间步执行时间。加速比收敛至 4.3 倍（表1）。执行时间的周期性变化与雷诺数的周期性变化相对应（见图9上方的曲线）。

表1：第 5.1.1 节仿真设置中各时间步组件的平均执行时间及其标准差。

步或部分步	执行时间 [毫秒]
不使用 AdaLED	
微观传播器	$\mathbf{969 \pm 104}$
使用 AdaLED	
微观传播器	1055 ± 113
AdaLED 的开销	42 ± 1
仅宏观步	4.6 ± 0.3
平均值	**339 ± 183 （加速比为 2.9 倍）**
平均值（最后 15k 个时间步）	**225 ± 79 (4.3 倍加速比)**

除了测试自适应性外，我们还测试在模拟前半段禁用微观到宏观转变，会如何影响后半段 MLM 的质量。具体而言，我们预期，推迟初始转换并留出更多训练时间，可能有助于提高模拟后期的准确度和宏观利用率。因此，对于每个 Re(t) 剖面，我们总共测试四种设置：不延迟的自适应设置（A；AdaLED 的默认行为）、不延迟的非自适应设置（B）、延迟的自适应设置（C）以及延迟的非自适应设置（D）。每种设置均使用不同的随机种子运行五次，以获得性能方差。

图 13 展示了第一个 Re(t) 剖面的宏观利用率以及速度的相对均方误差。我们观察到，在前半段仅进行训练并禁用转换（设置 D），相比其他设置，在模拟后半段实现了更高的准确度。事实上，与自适应设置 C 相比，非自适应设置 D 的宏观利用率更高、误差更低。这是意料之中的，因为运行前半段的训练数据集已经包含了有效预测后半段动力学所需的全部信息（两个剖面相似）。因此，无需进行在线训练。

然而，如图 14 所示，当系统状态区间随时间变化时，我们观察到不同的现象。在非自适应设置 B 和 D 中，系统进入此前未见过的 Re(t) $= 1200$ 状态区间时，宏观利用率会降至零；而自适应设置 A 和 C 最终能够适应，并达到 25% 至 35% 的宏观利用率。

这里需要指出，设置 D 和 C 可用的训练时间较长。设置 A 和 B 从一开始就加速了模拟，因此在模拟的前半段只有 6 小时用于训练；设置 D 和 C 则用 11 小时完成前半段，因此在后半段接受测试之前，训练时间几乎是前者的两倍。我们认为，这一现象是在线代理模型的一项重要特性：它们实现的加速比越高，可用于训练的时间就越少。

5.1.3. 多分辨率自编码器

本节通过消融研究分析多分辨率卷积自编码器的优势。我们比较三种情况：(i) 单分辨率；(ii) 在圆柱体周围采用 256×256 补丁的多分辨率；以及 (iii) 采用 224×224 补丁的多分辨率。每种情况下，我们使用随机阈值 $E_{\max}$ 和 $\sigma_{\max}$、学习率、力缩放 α_F、潜状态大小 d_z、CNN通道数和层数，分别进行 80 次运行。超参数搜索空间列于表 2。CNN架构、其余超参数以及训练执行时间的细分情况见附录 D.4。仿真设置与第 5.1.1 节相同（雷诺数 Re(t) 每 5000 个时间步在 600、750 和 900 之间循环），但运行时间较短，为 60 000 个时间步。平均而言，单次模拟运行约需 16 小时。我们考察三项性能指标：总宏观利用率 η、平均速度相对均方误差 E（式 (13)）以及平均力误差 E_F（式 (14)）。平均值仅包含仅宏观阶段的数据。

比较结果如图 15 和图 16 所示，其中宏观利用率分别相对于平均误差 E 和 E_F 进行绘制。我们观察到，与单分辨率自编码器相比，多分辨率方法在宏观利用率和准确度方面都具有明显优势。图中的圈选点对应于第 5.1.1 节和第 5.1.2 节中使用的超参数集，该超参数集

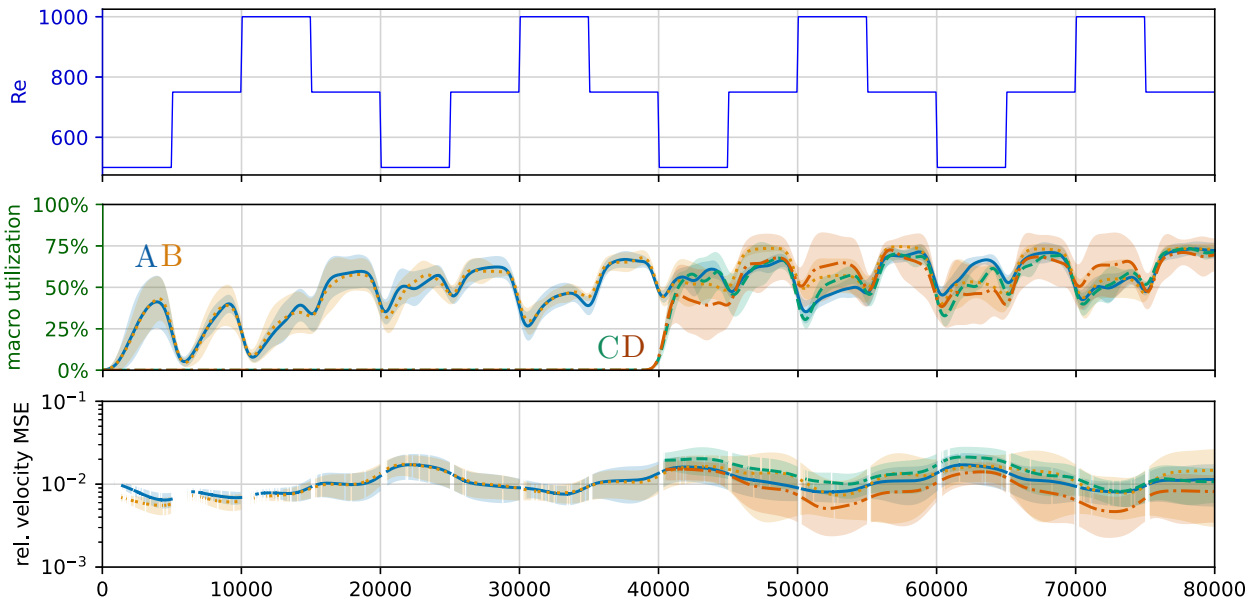

图 13：第 5.1.2 节中对雷诺数(t) 完全重复配置的自适应性和转换延迟研究。曲线表示：无延迟的自适应（A ）、无延迟的非自适应（B ）、有延迟的自适应（C ）、有延迟的非自适应（D ）。上：雷诺数剖面；中：宏观利用率 η；下：仅宏观阶段中速度的平滑相对均方误差。阴影区域（如有）表示相同设置重复运行五次所得结果的标准差。

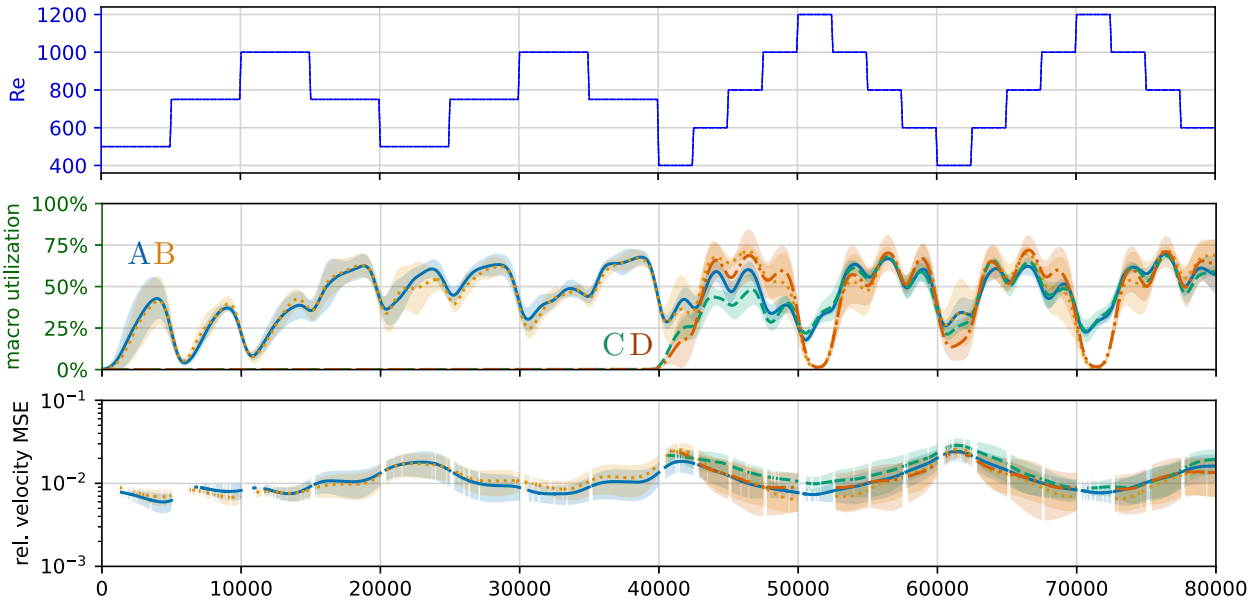

图 14：与图 13 类似，展示了 雷诺数(t) 在模拟的后半段改变剖面的情况。非自适应情形（B 和 D ）在后半段的性能下降清晰可见。

表 2：多分辨率自编码器研究的参数搜索空间及选定参数集。

参数	搜索空间		已选择
多分辨率?	no	yes	yes
内部分辨率	N/A	{256x256, 224x224}	224x224
CNN 层数	{5, 6}	{4, 5}	4
每层通道数	{16, 20, 24}		16
d_z（每种分辨率）	{4, 8, 12, 16, 24}		8
自编码器学习率	对数均匀分布(0.0001, 0.001)		0.00047
长短期记忆网络学习率	对数均匀分布(0.0003, 0.003)		0.00126
缩放 α_F	对数均匀分布(0.03, 30.0)		7.2
$E_{\max}$	对数均匀分布(0.001, 0.1)		0.017
$\sigma^2_{\max}$	对数均匀分布(0.00001, 0.1)		0.00035

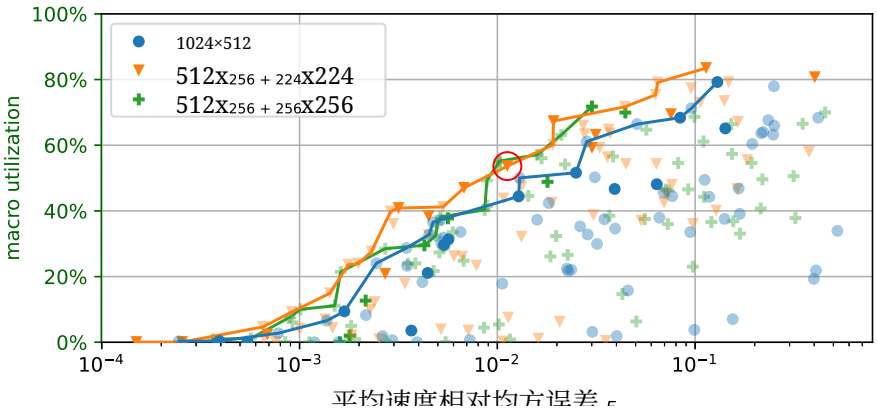

图15：多分辨率自编码器多目标研究（第 5.1.3 节），以优化总宏观利用率 η 和速度误差 E为目标。各条线表示每种自编码器配置的帕累托前沿。较深色的标记表示在 η–E_F 意义上也处于最优的样本（图16）。圈出的样本是本研究其余部分使用的参考参数集。

在 60000 个时间步内，宏观利用率达到 54%（运行 300 000 个时间步时为 69%）。该超参数集在不同雷诺数剖面上的性能见附录D.6。需要注意的是，结果会受到随机变化的影响：误差约变化 ±5%（相对值），宏观利用率约变化 ±3%（绝对值），具体取决于随机种子。

6. 讨论

我们提出了 AdaLED，这是一种框架，它采用卷积自编码器和循环神经网络长短期记忆网络集成，学习由数据驱动、在线、自适应的机器学习模型，以加速复杂系统的模拟。该模型与原始模拟（微观传播器）并行训练，并能在线适应新发现的动力学。更重要的是，该模型会监测自身的准确度和预测不确定性，并且只有在准确度较高且不确定性较低时才会取代微观传播器。这一机制使模拟能够在已学习的状态空间区域中加速，即使模型无法或尚未充分训练，不能忠实地再现整个复杂状态空间的动力学。对于状态中代表不足或不属于训练数据的区域，AdaLED 则使用原始计算求解器。

我们在三个基准问题中展示 AdaLED：参数非线性 $\mu \in [1, 3]$ 不断变化的范德波尔振子动力学、扩散参数 $d \in [0.1, 0.2]$ 不断变化的二维反应扩散方程，以及雷诺数 $\in [400, 1200]$ 不断变化时绕流圆柱体的流动。在这些基准问题中，我们展示了它逐步训练机器学习模型、仅在预测可靠时利用其预测，以及在系统进入相空间中的未见状态

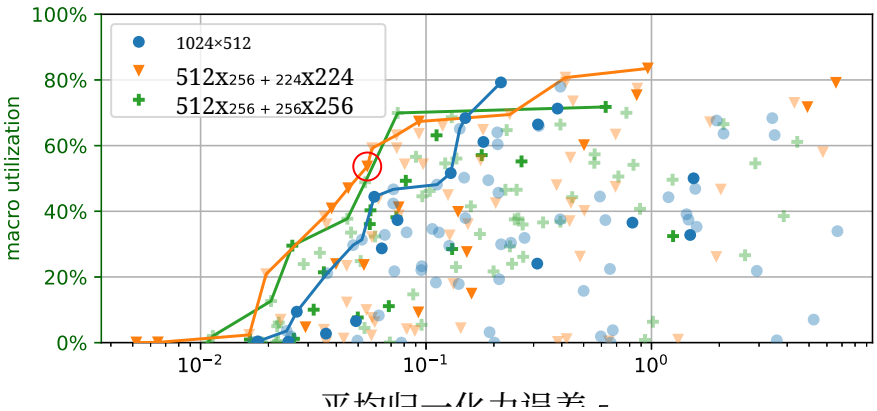

图16：与图15类似，其中 x 轴表示平均归一化圆柱体力均方根误差 E_F，而非速度场误差 E。此处，较深色的符号表示按照 η–E 标准选出的最优样本。

区间时进行检测的能力。训练后的模型在训练期间见过的所有动态状态下均表现出较高的准确度，且不会遭受灾难性遗忘。

对于雷诺数不断变化的圆柱绕流，AdaLED 从未训练网络开始，便能重现差异极大的动力学状态的动力学，在为期 3 天的模拟中实现 2.9 倍的净加速比。实现这一加速比的代价仅为：速度场上的均方误差为 ∼1%，圆柱体上的力的均方根误差为 ∼5%，互相关为 0.99。通过增大 $E_{\max}$ 和 $\sigma_{\max}$，可以进一步提高加速比，但准确度会降低。我们强调，与其他框架相比，AdaLED 能够更好地控制加速比与准确度之间的权衡。据我们所知，AdaLED 是首个能够使用单个代理模型，高效学习并推进复杂流动在不同状态区间下的高维动力学的方法，可在准确度与加速比之间实现稳健的权衡。

我们的研究结果表明，AdaLED 是一种强大的自适应算法，可用于自适应地构建和交互式使用代理模型，从而加速复杂的多尺度模拟。我们认为，AdaLED 可作为黑箱加速器，利用计算密集型任务中的重复模式。在未来，我们计划研究其在强化学习任务和模型参数优化中的加速能力，因为多个模拟可以共享同一个代理模型。

此外，我们认为，所提出的框架是对数字孪生文献的重要贡献 [63, 64]。AdaLED 将数据同化、实时监测和在线自适应数据驱动学习相结合，以构建代理模型。如果能够从任意初始条件出发，随时对物理系统进行模拟，则所提出的框架可直接适用。否则，只需进行 minor 修改即可应用（关闭微观尺度求解器的重启功能）。通过这种方式，该框架可用于学习物理系统的数字副本，即数字孪生。无需承担实验或完全解析模拟的成本和计算负担，也无需让原始系统暴露于不利条件的风险，即可随时测试代理模型在不同条件和参数化设置下的响应 [22, 65]。

在应用方面，AdaLED 可采用更先进的自编码器，例如变分自编码器 [66], 考虑时间相关性的自编码器 [67], ，或基于空间填充曲线 [68]和八叉树 [69, 70]的非均匀自编码器。目前的一项研究课题是利用后者，帮助 AdaLED 扩展至三维流动。此外，目前所有潜在状态变量都被视为同等重要。该方法可受益于能够评估各潜在维度对重建 [71]的重要性的压缩技术。

近期提出的分层深度学习时间步进器 [72], 潜在空间上的降阶传播器 [73], 以及 Autoformer 网络 [74] 在偏微分方程和其他复杂时间序列数据中展现出良好的效果。这些算法可用作高效的宏观传播器。宏观传播器的运行速度极快，为更先进的技术创造了空间，例如在强化

学习 [75]中规划最优动作。

同样，如果应用场景允许，我们可以提前模拟许多步，并查看未来的预测不确定性，以判断是否应提前进行宏观到微观转换，从而检测数据中代表性不足的动态状态。还可以训练一个额外的网络，根据当前潜在状态估计解码器重构误差。结合宏观传播器，宏观到微观转换的判据可以基于集成的联合不确定性以及这一重构误差。

最后，我们计划研究 AdaLED 周期的改进调度和精细控制，以及微观—宏观转换，以减少在微观尺度执行的时间步总数，并更好地控制自适应性与加速比之间的权衡。在这方面，AdaLED 可以受益于用于监督学习算法不确定性量化的新型算法 [76]。

7. 致谢

我们感谢欧洲高性能计算联合事业（EuroHPC）DCoMEX 项目拨款（956201-H2020-JTI-EuroHPC-2019-1）提供的支持，以及瑞士国家超级计算中心（CSCS）在 s930 和 s1160 项目下提供的计算资源。我们还要感谢 Pascal Weber（ETHZ）提供的多次有益讨论。

8. 数据与代码可用性

源代码将在论文发表后于 https://github.com/cselab/adaled 公开提供。

参考文献

[1] F. Lateef，基于模拟的学习：就像真实事物一样，《急诊、创伤与休克杂志》3 (4) (2010) 348.[2] V. Springel、S. D. White、A. Jenkins、C. S. Frenk、N. Yoshida、L. Gao、J. Navarro、R. Thacker、D. Croton、J. Helly 等，星系和类星体的形成、演化与聚集模拟，《自然》435 (7042) (2005) 629–636。[3] T. Kurth、S. Treichler、J. Romero、M. Mudigonda、N. Luehr、E. Phillips、A. Mahesh、M. Matheson、J. Deslippe、M. Fatica 等，面向气候分析的百亿亿次级深度学习，载于：SC18：国际高性能计算、网络、存储与分析会议，IEEE，2018，第649–660页。[4] O. Ghattas、K. Willcox，从数据中学习基于物理的模型：来自反问题与模型降阶的视角，《数值学报》30 (2021) 445–554。doi:10.1017/S0962492921000064。[5] W. Gong、Q. Duan、J. Li、C. Wang、Z. Di、Y. Dai、A. Ye、C. Miao，利用自适应代理建模对通用陆面模型进行多目标参数优化，《水文学与地球系统科学》19 (5) (2015) 2409–2425。[6] S. Verma、G. Novati、P. Koumoutsakos，通过深度强化学习利用涡旋实现高效的群体游动，《美国国家科学院院刊》115 (23) (2018) 5849–5854。[7] G. Novati、H. L. de Laroussilhe、P. Koumoutsakos，利用多智能体强化学习实现湍流建模自动化，《自然·机器智能》3 (1) (2021) 87–96。[8] S. S. Du、S. M. Kakade、R. Wang、L. F. Yang，良好的表征是否足以实现样本高效的强化学习？，arXiv 预印本 arXiv:1910.03016 (2019)。[9] M. Taufer、E. Deelman、R. F. d. Silva、T. Estrada、M. Hall、M. Livny，面向高通量应用的稳健科学路线图：开发者视角，载于：2021 IEEE 国际集群计算会议（CLUSTER），2021，第807–808页。doi:10.1109/Cluster48925.2021.00068。[10] I. G. Kevrekidis、C. W. Gear、J. M. Hyman、P. G. Kevrekidis、O. Runborg、C. Theodoropoulos 等，无方程粗粒化多尺度计算：使微观模拟器能够执行系统级分析，《数学科学通讯》1 (4) (2003) 715–762。[11] C. R. Laing、T. Frewen、I. G. Kevrekidis，双眼竞争的降阶模型，《计算神经科学杂志》28 (3) (2010) 459–476。[12] Y. Bar-Sinai、S. Hoyer、J. Hickey、M. P. Brenner，学习偏微分方程的数据驱动离散化方法，《美国国家科学院院刊》116 (31) (2019) 15344–15349。[13] E. Weinan、B. Engquist 等，异质多尺度方法，《数学科学通讯》1 (1) (2003) 87–132。[14] E. Weinan、B. Engquist、X. Li、W. Ren、E. Vanden-Eijnden，异质多尺度方法：综述，《计算物理通讯》2 (3) (2007) 367–450。

[15] M. Tao、H. Owhadi、J. E. Marsden，通过流平均实现具有隐藏慢动力学的刚性常微分方程、随机微分方程和哈密顿系统的非侵入式保结构多尺度积分，《多尺度建模与仿真》8 (4) (2010) 1269–1324。[16] J. N. Kutz、S. L. Brunton、B. W. Brunton、J. L. Proctor，《动态模态分解：复杂系统的数据驱动建模》，SIAM，2016。[17] R. R. Coifman、S. Lafon，扩散映射，《应用与计算调和分析》21 (1) (2006) 5–30。[18] P. R. Vlachas、J. Pathak、B. R. Hunt、T. P. Sapsis、M. Girvan、E. Ott、P. Koumoutsakos，用于复杂时空动力学预测的循环神经网络中的反向传播算法与储层计算，《神经网络》126 (2020) 191–217。[19] P. R. Vlachas、W. Byeon、Z. Y. Wan、T. P. Sapsis、P. Koumoutsakos，利用长短期记忆网络对高维混沌系统进行数据驱动预测，《英国皇家学会学报 A：数学、物理与工程科学》474 (2213) (2018) 20170844。[20] Z. Y. Wan、P. Vlachas、P. Koumoutsakos，复杂动力系统中极端事件的数据辅助降阶建模，《PLOS ONE》13 (5) (2018) e0197704。[21] S. Brunton、B. Noack、P. Koumoutsakos，流体力学中的机器学习，arXiv 预印本 arXiv:1905.11075 (2019)。[22] R. Vinuesa、S. L. Brunton，利用机器学习增强计算流体动力学，《自然·计算科学》2 (6) (2022) 358–366。[23] D. Kochkov、J. A. Smith、A. Alieva、Q. Wang、M. P. Brenner、S. Hoyer，机器学习加速的计算流体动力学，《美国国家科学院院刊》118 (21) (2021) e2101784118。[24] P. R. Vlachas、G. Arampatzis、C. Uhler、P. Koumoutsakos，通过学习复杂系统的有效动力学实现多尺度模拟，《自然·机器智能》4 (4) (2022) 359–366。[25] P. R. Vlachas，跨尺度复杂系统有效动力学的学习与预测，博士论文，苏黎世联邦理工学院 (2022)。[26] P. R. Vlachas、J. Zavadlav、M. Praprotnik、P. Koumoutsakos，通过学习有效动力学加速分子系统模拟，《化学理论与计算杂志》18 (1) (2021) 538–549。[27] T. Wu、T. Maruyama、J. Leskovec，通过潜在全局演化学习加速偏微分方程，arXiv 预印本 arXiv:2206.07681 (2022)。[28] S. Wiewel、M. Becher、N. Thuerey，潜在空间物理：迈向流体流动时间演化的学习，载于：《计算机图形论坛》，第38卷，Wiley Online Library，2019，第71–82页。[29] F. J. Gonzalez、M. Balajewicz，用于学习流体系统低维特征动力学的深度卷积循环自编码器，arXiv 预印本 arXiv:1808.01346 (2018)。[30] K. Fukami、K. Hasegawa、T. Nakamura、M. Morimoto、K. Fukagata，利用神经网络进行模型降阶：在层流和湍流中的应用，《SN计算机科学》2 (6) (2021) 1–16。[31] K. Stachenfeld、D. B. Fielding、D. Kochkov、M. Cranmer、T. Pfaff、J. Godwin、C. Cui、S. Ho、P. Battaglia、A. Sanchez-Gonzalez，用于高效湍流模拟的学习型粗粒化模型，arXiv 预印本 arXiv:2112.15275 (2021)。[32] N. Geneva、N. Zabaras，利用物理约束深度自回归网络对偏微分方程系统的动力学进行建模，《计算物理学杂志》403 (2020) 109056。[33] R. Maulik、B. Lusch、P. Balaprakash，利用循环神经网络和卷积自编码器对对流主导系统进行降阶建模，《流体物理学》33 (3) (2021) 037106。[34] K. Hasegawa、K. Fukami、T. Murata、K. Fukagata，基于机器学习的不同形状钝体周围非定常流动降阶建模，《理论与计算流体动力学》34 (4) (2020) 367–383。[35] P. Pant、R. Doshi、P. Bahl、A. Barati Farimani，用于流体模拟降阶建模和高效时间演化的深度学习，《流体物理学》33 (10) (2021) 107101。[36] H. Eivazi、H. Veisi、M. H. Naderi、V. Esfahanian，用于非定常流动非线性模型降阶的深度神经网络，《流体物理学》32 (10) (2020) 105104。[37] X. Zhang、T. Ji、F. Xie、H. Zheng、Y. Zheng，通过压缩感知降阶建模，利用稀疏测量预测非定常流动，《计算力学与工程方法》393 (2022) 114800。[38] T. Simpson、N. Dervilis、E. Chatzi，通过自编码器和长短期记忆网络实现非线性系统模型降阶的机器学习方法，arXiv 预印本 arXiv:2109.11213 (2021)。[39] P. Wu、J. Sun、X. Chang、W. Zhang、R. Arcucci、Y. Guo、C. C. Pain，利用时间卷积神经网络构建数据驱动的降阶模型，《计算力学与工程方法》360 (2020) 112766。[40] S. Fresca、A. Manzoni，POD-DL-ROM：通过本征正交分解增强用于非线性参数化偏微分方程的基于深度学习的降阶模型，《计算力学与工程方法》388 (2022) 114181。[41] B. Peherstorfer、K. Willcox，面向非侵入式基于投影的模型降阶的数据驱动算子推断，《计算力学与工程方法》306 (2016) 196–215。[42] P. Benner、M. Ohlberger、A. Cohen、K. Willcox，模型降阶与近似：理论与算法，SIAM，2017。[43] D. Galbally、K. Fidkowski、K. Willcox、O. Ghattas，大规模反问题中不确定性量化的非线性模型降阶，《国际工程数值方法杂志》81 (12) (2010) 1581–1608。[44] K. Vlachas、K. Tatsis、K. Agathos、A. R. Brink、E. Chatzi，非线性参数化模型降阶的局部基近似方法，《声学与振动杂志》502 (2021) 116055。[45] K. Vlachas、K. Tatsis、K. Agathos、A. R. Brink、D. Quinn、E. Chatzi，用于局部非线性特征纳入的参数化模型降阶，载于：《非线性动力学进展》，Springer，2022，第373–383页。[46] W. D. Fries、X. He、Y. Choi，LaSDI：参数化潜在空间动力学识别，《计算力学与工程方法》399 (2022) 115436。[47] X. He、Y. Choi、W. D. Fries、J. Belof、J.-S. Chen，gLaSDI：参数化物理信息贪心潜在空间动力学。

辨识，arXiv预印本 arXiv:2204.12005（2022）。[48] B. Peherstorfer、K. Willcox，动态数据驱动降阶模型，《计算力学与工程方法》291（2015）21–41。[49] B. Peherstorfer、K. Willcox，通过低秩更新实现非线性系统的在线自适应模型降阶，《SIAM科学计算期刊》37（4）（2015）A2123–A2150。[50] Q. Wang、O. Fink、L. Van Gool、D. Dai，持续测试时域适应，载于：IEEE/CVF计算机视觉与模式识别会议论文集，2022，第7201–7211页。[51] P. J. Werbos，反向传播的推广及其在循环天然气市场模型中的应用，《神经网络》1（4）（1988）339–356。[52] J. Kirkpatrick、R. Pascanu、N. Rabinowitz、J. Veness、G. Desjardins、A. A. Rusu、K. Milan、J. Quan、T. Ramalho、A. Grabska-Barwinska等，克服神经网络中的灾难性遗忘，《美国国家科学院院刊》114（13）（2017）3521–3526。[53] B. Lakshminarayanan、A. Pritzel、C. Blundell，使用深度集成实现简单且可扩展的预测不确定性估计，arXiv预印本 arXiv:1612.01474（2016）。[54] D. A. Nix、A. S. Weigend，估计目标概率分布的均值和方差，载于：1994年IEEE国际神经网络会议（ICNN’94）论文集，第1卷，IEEE，1994，第55–60页。[55] B. Van der Pol，论“弛豫振荡”，《伦敦、爱丁堡和都柏林哲学杂志与科学期刊》2（11）（1926）978–992。[56] D. Kaplan、L. Glass，《理解非线性动力学》，Springer Science & Business Media，1997。[57] K. Champion、B. Lusch、J. N. Kutz、S. L. Brunton，数据驱动的坐标与控制方程发现，《美国国家科学院院刊》116（45）（2019）22445–22451。[58] D. Floryan、M. D. Graham，数据驱动的内在动力学发现，《自然·机器智能》4（12）（2022）1113–1120。[59] P. Angot、C.-H. Bruneau、P. Fabrie，一种将障碍物纳入不可压缩黏性流计算的惩罚法，《数值数学》81（4）（1999）497–520。[60] A. J. Chorin，纳维-斯托克斯方程的数值求解，《计算数学》22（104）（1968）745–762。[61] A. T. Mohan、N. Lubbers、D. Livescu、M. Chertkov，在三维湍流的神经网络粗粒化中嵌入严格物理约束，arXiv预印本 arXiv:2002.00021（2020）。[62] M. Chatzimanolakis、P. Weber、P. Koumoutsakos，CubismAMR——一种用于分布式块结构自适应网格细化的 C++ 库，arXiv预印本 arXiv:2206.07345（2022）。[63] A. Rasheed、O. San、T. Kvamsdal，数字孪生：从建模角度看其价值、挑战与促成因素，《IEEE Access》8（2020）21980–22012。[64] M. G. Kapteyn、J. V. Pretorius、K. E. Willcox，为大规模预测型数字孪生提供支持的概率图模型基础，《自然·计算科学》1（5）（2021）337–347。[65] R. Vinuesa、S. L. Brunton、B. J. McKeon，机器学习对流体力学实验的变革潜力，arXiv预印本 arXiv:2303.15832（2023）。[66] D. P. Kingma、M. Welling，自编码变分贝叶斯，arXiv预印本 arXiv:1312.6114（2013）。[67] L. Girin、S. Leglaive、X. Bie、J. Diard、T. Hueber、X. Alameda-Pineda，动力学变分自编码器：全面综述，arXiv预印本 arXiv:2008.12595（2020）。[68] C. E. Heaney、Y. Li、O. K. Matar、C. C. Pain，将卷积神经网络应用于基于空间填充曲线的非结构网格数据，arXiv预印本 arXiv:2011.14820（2020）。[69] G. Riegler、A. Osman Ulusoy、A. Geiger，Octnet：学习高分辨率的深度三维表示，载于：IEEE计算机视觉与模式识别会议论文集，2017，第3577–3586页。[70] M. Tatarchenko、A. Dosovitskiy、T. Brox，八叉树生成网络：用于高分辨率三维输出的高效卷积架构，载于：IEEE国际计算机视觉会议论文集，2017，第2088–2096页。[71] K. Fukami、T. Nakamura、K. Fukagata，基于卷积神经网络的分层自编码器，用于流场数据的非线性模态分解，《流体物理学》32（9）（2020）095110。[72] Y. Liu、J. N. Kutz、S. L. Brunton，多尺度微分方程时间步进器的分层深度学习，《皇家学会哲学汇刊A辑》380（2229）（2022）20210200。[73] S. Pawar、S. Rahman、H. Vaddireddy、O. San、A. Rasheed、P. Vedula，一种用于流体流动非侵入式降阶建模的深度学习工具，《流体物理学》31（8）（2019）085101。[74] H. Wu、J. Xu、J. Wang、M. Long，Autoformer：用于长期序列预测的自相关分解式Transformer，《神经信息处理系统进展》34（2021）22419–22430。[75] T. M. Moerland、J. Broekens、C. M. Jonker，基于模型的强化学习：综述，arXiv预印本 arXiv:2006.16712（2020）。[76] R. Egele、R. Maulik、K. Raghavan、B. Lusch、I. Guyon、P. Balaprakash，Autodeuq：一种用于不确定性量化的自动化深度集成，载于：2022年第26届国际模式识别会议（ICPR）论文集，IEEE，2022，第1908–1914页。[77] S. Hochreiter、J. Schmidhuber，长短期记忆，《神经计算》9（8）（1997）1735–1780。[78] D. P. Kingma、J. Ba，Adam：一种随机优化方法，arXiv预印本 arXiv:1412.6980（2014）。[79] I. J. Goodfellow、J. Shlens、C. Szegedy，解释并利用对抗样本，arXiv预印本 arXiv:1412.6572（2014）。[80] W. Jakob、J. Rhinelander、D. Moldovan，pybind11——C++11 与Python之间的无缝互操作，https://github.com/pybind/pybind11（2017）。

A. 宏观传播器长短期记忆网络

AdaLED 的宏观传播器是由多层长短期记忆网络（LSTM） [77]循环神经网络（RNN）组成的集成。给定输入状态 $\boldsymbol{\xi}_t \in \mathbb{R}^{d_\xi}$、当前隐状态$\mathbf{h}_t \in \mathbb{R}^{d_h}$ 和细胞状态$\mathbf{c}_t \in \mathbb{R}^{d_h}$，每个网络的每一层都会计算下一个隐状态$\mathbf{h}_{t+\Delta t}$ 和细胞状态$\mathbf{c}_{t+\Delta t}$ ，如下所示（为简洁起见，省略层和集成的符号）：

$$\begin{aligned}
\mathbf{i}_{t+\Delta t} &= \sigma(\mathbf{W}_i[\boldsymbol{\xi}_t, \mathbf{h}_t] + \mathbf{b}_i),\\
\mathbf{f}_{t+\Delta t} &= \sigma(\mathbf{W}_f[\boldsymbol{\xi}_t, \mathbf{h}_t] + \mathbf{b}_f),\\
\mathbf{g}_{t+\Delta t} &= \tanh(\mathbf{W}_g[\boldsymbol{\xi}_t, \mathbf{h}_t] + \mathbf{b}_g),\\
\mathbf{o}_{t+\Delta t} &= \sigma(\mathbf{W}_o[\boldsymbol{\xi}_t, \mathbf{h}_t] + \mathbf{b}_o),\\
\mathbf{c}_{t+\Delta t} &= \mathbf{f}_{t+\Delta t} \odot \mathbf{c}_t + \mathbf{i}_{t+\Delta t} \odot \mathbf{g}_{t+\Delta t},\\
\mathbf{h}_{t+\Delta t} &= \mathbf{o}_{t+\Delta t} \odot \tanh(\mathbf{c}_{t+\Delta t}),
\end{aligned} \tag{A.1}$$

其中，$\mathbf{i}_t$、$\mathbf{f}_t$、$\mathbf{g}_t$和$\mathbf{o}_t$分别为输入门、遗忘门、细胞门和输出门。矩阵$\mathbf{W}_i$、$\mathbf{W}_f$、$\mathbf{W}_g$、$\mathbf{W}_o \in \mathbb{R}^{d_h \times (d_h + d_z)}$以及向量$\mathbf{b}_i$、$\mathbf{b}_f$、$\mathbf{b}_g$、$\mathbf{b}_o \in \mathbb{R}^{d_h}$均为可训练参数。方括号 $[\dots]$ 表示拼接， σ 表示 Sigmoid 函数， $\odot$ 表示逐元素乘法。

第一层的输入状态 $\boldsymbol{\xi}_t^{(1)}$ 等于 $\boldsymbol{\xi}_t^{(1)} = [\mathbf{z}_t, \mathbf{q}_t, \mathbf{f}_t]$，其中 $\mathbf{z}_t \in \mathbb{R}^{d_z}$ 是潜在状态，$\mathbf{q}_t \in \mathbb{R}^{d_q}$ 是关注量，$\mathbf{f}_t \in \mathbb{R}^{d_F}$ 是外部强迫。在范德波尔振子（VdP）研究中，$\mathbf{z}_t = \mathbf{z}(t) = [x(t)\ y(t)],$，$\mathbf{f}_t = f(t) = \mu(t)$。在反应-扩散案例研究中，$\mathbf{z}_t$是自编码器的输出，且 $f_t = d(t)$。在计算流体力学研究中，$\mathbf{z}(t)$ 是（多分辨率）自编码器的输出，$\mathbf{q}(t) = \alpha_F \mathbf{F}^{\mathrm{cyl}}(t)$ 是缩放后的圆柱体上的力， $f(t)$ 是归一化的雷诺数$\sim f(t) = \mathrm{Re}(t) = 2.4\,\mathrm{Re}(t)/\mathrm{Re}_{\max} - 1.2$。在 $\mathrm{Re} \le 1000$、$\mathrm{Re}_{\max} = 1000$ 的研究中，以及 Re 最大达到 1200 的研究中，$\mathrm{Re}_{\max} = 1200$。对于其他层 $l \ge 2$，输入状态 $\boldsymbol{\xi}_t^{(l)}$ 等于前一层的隐状态 $\mathbf{h}_{t+\Delta t}^{(l-1)}$ 。

残 差 $\Delta\mathbf{w}(t) = \mathbf{w}(t+\Delta t) - \mathbf{w}(t)$、$\mathbf{w}(t) = [\mathbf{z}(t),\ \mathbf{q}(t)]$ 和不确定性 $\boldsymbol{\sigma}_{\Delta w}(t)$ 已给出 b y

$$\begin{aligned}
\Delta\mathbf{w}(t) &= \mathbf{W}_w \mathbf{h}_{t+\Delta t}^{(L)} + \mathbf{b}_w,\\
\boldsymbol{\sigma}_{\Delta\mathbf{w}}^2(t) &= \mathrm{SoftPlus}_\epsilon(\mathbf{W}'_\sigma\, \mathrm{CELU}(\mathbf{W}_\sigma \mathbf{h}_{t+\Delta t}^{(L)} + \mathbf{b}_\sigma) + \mathbf{b}'_\sigma),
\end{aligned} \tag{A.2}$$

其中，$\mathbf{h}^{(L)}$ 是最终层的隐状态 L。矩阵 $\mathbf{W}_w \in \mathbb{R}^{(d_z+d_q)\times d_h}$、$\mathbf{W}_w \in \mathbb{R}^{d_\sigma \times d_h}$ 和 $\mathbf{W}'_\sigma \in \mathbb{R}^{(d_z+d_q)\times d_\sigma}$，以及偏置 $\mathbf{b}_w$、$\mathbf{b}'_\sigma \in \mathbb{R}^{d_z+d_q}$和 $\mathbf{b}_\sigma \in \mathbb{R}^{d_\sigma}$（含 $d_\sigma = 100$）是可训练参数。函数 $\mathrm{CELU}(x) = \text{最大值}(0, x) + \text{最小值}(0, \exp(x) - 1)$ 和 $\mathrm{SoftPlus}_\epsilon(x) = \log(1 + \exp(x)) + \epsilon$（含 $\epsilon = 10^{-6}$）是非线性函数。 $\boldsymbol{\sigma}_{\Delta\mathbf{w}}$ 的参数与其余参数分开训练，具体说明见第 2.3 节。具体而言，参数 $\mathbf{W}_\sigma$、$\mathbf{W}'_\sigma$、$\mathbf{b}_\sigma$ 和 $\mathbf{b}'_\sigma$ 构成可训练参数 $\boldsymbol{\theta}_\sigma$，而其他参数构成 $\boldsymbol{\theta}_\mu$。

在集成中，每个 LSTM 网络都加入了用于计算 $\Delta\mathbf{w}(t)$ 和$\boldsymbol{\sigma}_{\Delta w}^2(t)$ 的附加层，并使用各自的可训练参数，从各自随机初始化的数值开始单独训练。训练采用 Adam 优化器 [78] 和时间反向传播 [51] ，以最小化第 2.3 节所述逐状态损失的轨迹总和。最后，在 VdP 研究中，使用了 $L = 3$ 层，每层的隐状态大小为 $d_h = 32$ ，总计包含 8176 个可训练参数。在反应-扩散案例研究中，使用了 $L = 2$ 层和 $d_h = 64$ ，参数总数为 60 308（针对 $d_z = 8$）。在计算流体力学案例研究中，使用了 $L = 2$ 层和 $d_h = 32$ ，参数总数为 20 944（针对 $d_z = 16$）。

B. 范德波尔振子案例研究的详细信息

在给定固定的 μ 时，范德波尔振子会进入一个极限环。不同 μ 值对应的极限环如图 B.1 所示。

$\mu\,(t)$ 的三个案例详情如下。在 $\mu_{\mathrm{ALT}}\,(t)$ 案例中， μ 每隔 50000 个时间步便在$\mu = 1$ 和 $\mu = 3$ 之间交替变化。在 $\mu_{\mathrm{RAND}}\,(t)$ 案例中， μ 在 1.96、1.23、2.80 这些值之间变化，

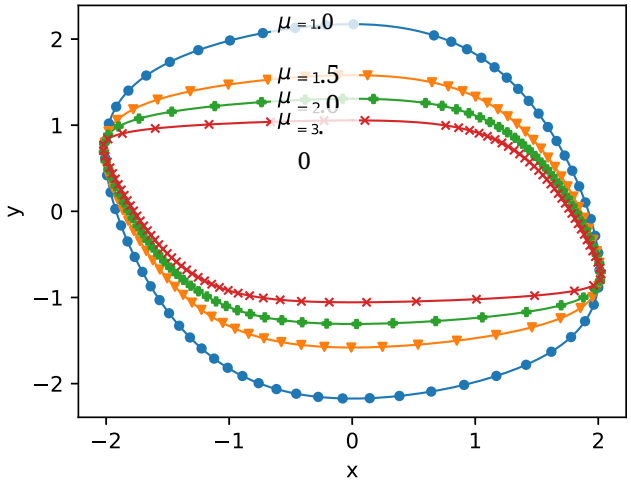

图 B.1：范德波尔振子的极限环。

表 B.1：范德波尔振子的超参数研究及最终参数（以**粗体**显示）。右侧的百分比表示达到的最高宏观利用率。

参数	搜索空间	注释
学习率	对数均匀分布$(0.0002, 0.02)$	**0.002**
批量大小	$\{8, \mathbf{16}, 32, 64\}$	
LSTM隐藏状态大小 d_h	$\{8, 16, \mathbf{32}\}$	
LSTM层数 L	$\{1, 2, \mathbf{3}, 4\}$	
对抗训练?	**{否，** 是}	无显著影响
$\mu(t)$ 作为输入的一部分?	{否，**是}**	∼54% 与 ∼70%

2.34、1.61、2.57、1.49、3.00、1.69 和 1.00，其速率与 $\mu_{\mathrm{ALT}}(t)$ 相同。最后，通过平滑 $\mu_{\mathrm{RAND}}(t)$并向其中添加布朗噪声，计算出 $\mu_{\mathrm{BROWN}}(t)$ 剖面：

$$\begin{aligned}&\mu_{\mathrm{BROWN}}(t) = \mu_{\mathrm{RAND}}(t) + \alpha(\mu_{\mathrm{BROWN}}(t-\Delta t) - \mu_{\mathrm{RAND}}(t)) + \beta\epsilon(t), \quad \epsilon(t) \sim \mathcal{U}([-1,1]),\\ &\mu_{\mathrm{BROWN}}(0) = \mu_{\mathrm{RAND}}(0),\end{aligned} \tag{B.1}$$

其中 $\alpha = \exp(-\Delta t/200)$ $\Delta t = 0.1$ 和 $\beta = 0.005$。

B.1. 超参数研究

超参数的搜索空间及其选定值见表 B.1。本研究从所列范围内随机选取了 1536 组超参数值样本，以总宏观利用率和平均在线验证误差 E为优化目标。每次模拟运行 200 000 个时间步，其中 μ 每 25 000 个时间步在 1.5 和 2.5 之间交替变化。使用了 $E_{\max} = 0.14$ 和$\sigma_{\max} = 0.14$ 阈值。取得的最高宏观利用率为 70%。最终的超参数集（表 B.1）选取了一个帕累托最优样本，其宏观利用率为 68%，平均在线验证误差 E 为 0。008。我们还探索了对抗训练 [79, 53], ，但它对结果没有明显影响。此外，我们测试了将 $\mu(t)$作为输入的一部分时，网络的性能是否显著优于无法获取 $\mu(t)$的情况。结果表明，即使不向网络提供 $\mu(t)$，宏观传播器的接受率和总利用率仍然很高（∼54%）。不过，在这种情况下，宏观传播器预测的不确定性自然不会对仅宏观阶段中 $\mu(t)$的变化做出响应。

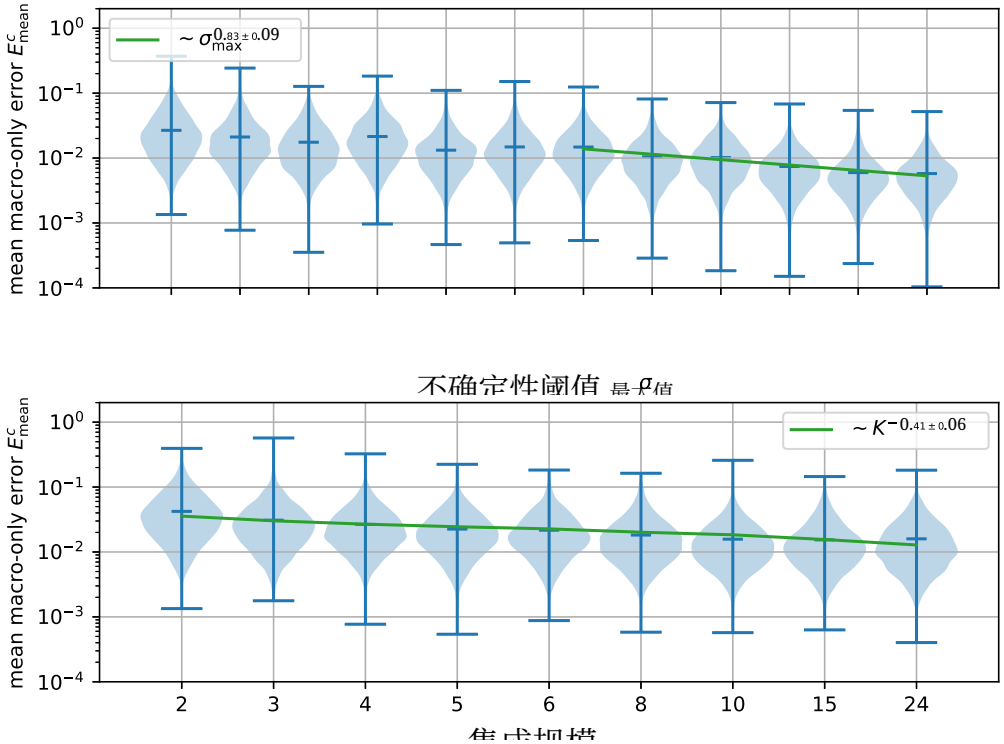

图 B.2：范德波尔振子案例研究（附录 B.2）中，预测误差与不确定性阈值 $\sigma_{\max}$ （上图）及集成规模 K （下图）的关系。每个小提琴图代表一次运行，展示了沿 AdaLED 周期的仅宏观预测误差均值分布。

B.2. 误差与阈值及集成规模的关系

通过采用更严格的不确定性阈值 $\sigma_{\max}$ 或增大集成规模 K，可以降低测试误差 E。为分析它们对 E 的影响程度，我们针对不同的 $\sigma_{\max}$ 和 K运行 $\mu_{\text{ALT}}(t)$案例。阈值 $E_{\max}$ 固定为 0.1。图 B.2 上方的图展示了不同 $\sigma_{\max}$ 和 K下的平均周期预测误差 $E^c_{\max}$ 分布（即一个循环 c中仅宏观步数上的平均误差）。我们注意到，当 $\sigma_{\max}$足够小时，误差随 $\sigma_{\max}$近似线性下降。这一趋势可以通过量纲分析来解释。具体而言， E 和 $\sigma_{\max}$ 的单位相同。图 B.2 下方的图显示， E 近似按$1/\sqrt{K}$的速率下降，这与中心极限定理相符。具体来说，假设每个长短期记忆网络都能在零偏差噪声的范围内生成正确轨迹，那么噪声就会以 $1/\sqrt{K}$的速率相互抵消。

不过需要注意的是，虽然更严格的 $\sigma_{\max}$ 能提高精度，但会降低宏观利用率η。例如，对于 $\sigma_{\max} = 0.1$，宏观利用率 η 分别为 60%、29% 和 18%，对应的数值为 0.02 和 0.01。同样，在训练轮次数固定的情况下，将集成规模从 $K = 5$ 提高到 $K = 20$ 可使误差降低 ∼2倍，但会使总训练时间增加 4 倍。因此，根据具体情况和目标，降低 $\sigma_{\max}$ 并提高 K 未必总是有利。

C. 反应-扩散研究的详细信息

系统的初始条件为 [57]

$$
\begin{aligned}
u(x, y, 0) &= \tanh\left(\sqrt{x^2+y^2}\cos\left(\operatorname{atan2}(y, x) - \sqrt{x^2+y^2}\right)\right), \\
v(x, y, 0) &= \tanh\left(\sqrt{x^2+y^2}\sin\left(\operatorname{atan2}(y, x) - \sqrt{x^2+y^2}\right)\right).
\end{aligned}
\tag{C.1}
$$

方程（10）采用四阶龙格-库塔-费尔伯格积分格式进行时间积分。扩散项采用满足零值冯·诺依曼边界条件的二阶中心差分模板。

数据集容量设为 1024 条轨迹，每条包含 24 个时间步。超参数包括潜在空间维度、RNN隐状态大小、层数、批量大小、学习率、每个 AdaLED 周期的训练量，以及阈值 $E_{\max}$ 和 $\sigma_{\max}$ ，均通过手动调优确定。自编码器由编码器和解码器组成，两者均为每层 16 个通道的 4 层卷积神经网络。自编码器架构见表C.1。数值（$\mathbf{u}$,$\mathbf{v}$）的取值范围为 $[-1,1]$，进入编码器前会缩小 1.1 倍，并在解码器末端恢复原大小。自编码器和循环神经网络均采用 0.001 的学习率和 64 的批量大小。训练以部分轮次的形式进行：每个 AdaLED 周期后，自编码器使用数据集中 6.25% 的状态进行训练，循环神经网络则使用已存储轨迹的 12.5% 进行训练。

D. 圆柱体尾流研究的详细信息

为触发涡脱落，我们在模拟开始时加入一段短暂的破坏对称性的垂直运动：

$$u_y^s(t) = e^{-\alpha t}\sin(\beta t)u_{y0}^s, \tag{D.1}$$

其中 $\alpha = 100$、 $\beta = 200$ 和 $u_{y0}^s = 0.05d$。

模拟使用 CubismAMR [62], 这一用于求解不可压缩纳维–斯托克斯方程的自适应网格细化（AMR）CPU–GPU 混合 C++代码执行。为了在 AdaLED 中以及通过 Python 使用该代码，我们添加了 Python绑定 [80], API，用于控制模拟的执行，并添加了用于导出和导入状态的 API。导出和导入状态时，沿用了现有的粗细网格AMR插值格式，将其作为均匀网格处理。

流体施加在圆柱体上的力 $\mathbf{F}^{\text{cyl}}$ 等于压力和黏性力之和，由 CubismAMR 提供：

$$\begin{aligned}
\mathbf{F}_{\text{cyl}} &= \mathbf{F}_{\text{p}} + \mathbf{F}_{\text{v}},\\
\mathbf{F}_{\text{p}} &= \oiint -p\mathbf{n}\,\mathrm{d}S\,,\\
\mathbf{F}_{\text{v}} &= \nu\rho \oiint (\nabla\mathbf{u} + \nabla\mathbf{u}^{\intercal})\cdot\mathbf{n}\,\mathrm{d}S\,,
\end{aligned} \tag{D.2}$$

其中 S 是圆柱体的表面，$\mathbf{n}$ 是外法向量。

D.1. 用于计算流体力学状态的自编码器

为了实现高加速比，宏观传播器并不直接作用于高维微观状态$\mathbf{u} \in \mathbb{R}^{d_v}$, $d_v = 512 \times 1024 \times 2 \approx 10^6$，而是作用于维度更低的潜在状态$\mathbf{z} \in \mathbb{R}^{d_z}$，其中$d_z \sim 10$。这一假设认为，这种转换确实可行：当我们需要较高分辨率来准确模拟流动动力学并获取精确的力$\mathbf{F}^{\text{cyl}}$时，动力学的实际维度可能很低。

为将速度场 $\mathbf{u}$ 压缩为潜在状态 $\mathbf{z}$，我们采用基于卷积神经网络（CNN）的自编码器。我们并不只是利用 $\mathbf{u}$的简单（相对）均方误差（MSE），让自编码器对 $\mathbf{u}$ 进行直接重建，而是将流体动力系统的特性纳入考虑：(i) 空间导数和涡量 $\omega = (\nabla \times \mathbf{u})_z$ 也是物理上至关重要的量（见公式（12））；(ii) 流动不可压缩，因此散度必须为零（$\nabla \cdot \mathbf{u} = 0$）；(iii) 我们假设，与圆柱体周围的流动相比，远离圆柱体处的流动对重建精度的要求较低。

如果自编码器仅针对最小化$\mathbf{u}$ 的均方误差（MSE）进行训练，而忽略导数的取值，那么重建得到的$\mathbf{u}$会带有较强的空间噪声，导致局部导数不准确，并使局部涡量偏高 $\omega = \omega_z = (\nabla \times \mathbf{u})_z$。这种噪声涡量会导致本研究所用的计算流体力学求解器进行不必要的网格加密 [62], ；该求解器采用自适应非均匀网格，并以局部涡量的大小作为网格加密和粗化准则。我们在损失函数中加入相对涡量重建误差 L_1 ，以确保重建后的涡量在原本较低的区域仍保持较低。这有助于减小网格尺寸

表C.1：反应-扩散案例研究中卷积自编码器的架构。所有卷积层均使用填充_模式=复制填充。批归一化层使用 PyTorch 的默认参数。

ID	形状	层
	$2 \times 96 \times 96$	输入
1	$16 \times 96 \times 96$	卷积(2, 16, 内核_大小=5, 填充=2)
1	$16 \times 96 \times 96$	批归一化()
2	$16 \times 48 \times 48$	平均池化(kernel_size=2, stride=2)
3	$16 \times 48 \times 48$	CELU()
4	$16 \times 48 \times 48$	卷积(16, 16, kernel_size=5, 填充=2)
1	$16 \times 48 \times 48$	批归一化()
5	$16 \times 24 \times 24$	平均池化(kernel_size=2, stride=2)
6	$16 \times 24 \times 24$	CELU()
7	$16 \times 24 \times 24$	卷积(16, 16, 内核_尺寸=5, 填充=2)
1	$16 \times 24 \times 24$	批归一化()
8	$16 \times 12 \times 12$	平均池化(内核_尺寸=2, 步幅=2)
9	$16 \times 12 \times 12$	CELU()
13	$16 \times 12 \times 12$	卷积(16, 16, 内核_尺寸=5, 填充=2)
14	$16 \times 6 \times 6$	平均池化(内核_尺寸=2, 步幅=2)
15	$16 \times 6 \times 6$	CELU()
16	576	展平()
17	$d_z^{(1)} = 8$	线性层()
18	$d_z^{(1)} = 8$	双曲正切函数()
	$d_z^{(1)} = 8$	$\mathbf{z}^{(i)}$
1	576	线性层()
2	$16 \times 6 \times 6$	视图层()
3	$16 \times 12 \times 12$	上采样(scale_factor=2.0, mode=双线性))
4	$16 \times 12 \times 12$	卷积(16, 16, kernel_size=3, 填充=1)
1	$16 \times 12 \times 12$	批归一化()
8	$16 \times 12 \times 12$	CELU()
9	$16 \times 24 \times 24$	上采样(scale_factor=2.0, mode=双线性))
10	$16 \times 24 \times 24$	卷积(16, 16, kernel_size=5, 填充=2)
1	$16 \times 24 \times 24$	批归一化()
11	$16 \times 24 \times 24$	CELU()
12	$16 \times 48 \times 48$	上采样(scale_factor=2.0, mode=双线性))
13	$16 \times 48 \times 48$	卷积(16, 16, kernel_size=5, 填充=2)
1	$16 \times 48 \times 48$	批归一化()
14	$16 \times 48 \times 48$	CELU()
15	$16 \times 96 \times 96$	上采样(scale_factor=2.0, mode=双线性))
16	$2 \times 96 \times 96$	卷积(16, 2, kernel_size=5, 填充=2)
17	$2 \times 96 \times 96$	双曲正切函数()
	50K	参数总数

与未考虑涡量损失的情况相比，性能下降约 10%–15%，并减轻了性能退化，从而避免部分抵消 AdaLED 所带来的优势。

通过预测流函数ψ 而非速度场 $\mathbf{u}$ [61]，可以将非零散度 $\nabla\cdot\mathbf{u}$ 完全作为硬约束加以避免。速度场 $\mathbf{u}$ 随后表示为：

$$u_x = \frac{\partial \psi}{\partial y}, \quad u_y = -\frac{\partial \psi}{\partial x}. \tag{D.3}$$

在离散形式下，使用2^{nd} 阶精度中心差分模板计算由 ψ 得到$\mathbf{u}$ 以及由$\mathbf{u}$ 得到 ω 的导数。例如，对于场 f 和网格间距 Δx， x方向的导数为：

$$\left.\frac{\partial f}{\partial x}\right|_{ij} = \frac{f_{i,j+1} - f_{i,j-1}}{2\Delta x} + \mathcal{O}(\Delta x^2). \tag{D.4}$$

因此，求导会使每个维度的两侧各减少一个单元格。为此， ψ 使用一个单元格的填充进行预测。

最后，我们希望优先降低圆柱体附近的重构损失，因为该区域会影响力 $\mathbf{F}_{\text{cyl}}$ ，而且此处的任何误差都会传播到流动的其他区域。此外，圆柱体后方的涡街相对平滑，不需要高分辨率。我们利用这两点观察，采用包含两对编码器—解码器对的多分辨率自编码器：一对用于整个以半分辨率下采样的计算域（$\mathbf{u}^{(1)}$），另一对用于聚焦圆柱体周围、保持全分辨率的子域（$\mathbf{u}^{(2)}$）。这两对彼此独立地运行，其压缩后的潜在状态 $\mathbf{z}^{(k)} \in \mathbb{R}^{d_z^{(k)}}$ 会拼接成最终的潜在状态 $\mathbf{z} \in \mathbb{R}^{d_z}$, $d_z = d_z^{(1)} + d_z^{(2)}$。具体细节将在下一节说明。

D.2. 多分辨率自编码器

在为二维（或三维）数组构建自编码器时，如果流动的不同部分呈现出不同的特征，且并非所有部分都需要相同的准确度，那么将多个在不同空间分辨率下运行的自编码器组合成一个自编码器，可能会带来益处。这样可以降低内存需求，提高计算效率和训练准确度（处理速度的提升会对准确度产生积极影响，专用架构也可能带来额外益处）。本节介绍如何构建此类多分辨率自编码器 。为简单起见，我们重点讨论使用两个编码器—解码器对重建二维标量数组 $\boldsymbol{\phi} \in \mathbb{R}^{H\times W}$ 的自编码器。此方法可以轻松推广到向量数组、两个以上的编码器—解码器对，以及更高维的数组。

对于每个编码器—解码器对 AE_k, $k \in \{1, 2\}$，我们定义一种下采样操作 $\mathbf{D}^{(k)}$ ： $\mathbb{R}^{H\times W} \to \mathbb{R}^{H^{(k)}\times W^{(k)}}$ ，将全分辨率数组 $\boldsymbol{\phi}$ 转换为较小的数组 $\boldsymbol{\phi}^{(k)} = \mathbf{D}^{(k)}(\boldsymbol{\phi})$，供 AE_k 处理。在本案例研究中，AE_1 用于以一半的分辨率重建整个区域（$H^{(1)} = H/2, W^{(1)} = W/2$），而 AE_2 则用于处理圆柱体周围尺寸为 $H^{(2)} \times W^{(2)}$ 的细节部分。因此，函数 $\mathbf{D}^{(1)}$ 和 $\mathbf{D}^{(2)}$ 定义如下：

$$\begin{aligned} D_{ij}^{(1)}(\boldsymbol{\phi}) &= \frac{1}{4}\left(\boldsymbol{\phi}_{2i,2j} + \boldsymbol{\phi}_{2i,2j+1} + \boldsymbol{\phi}_{2i+1,2j} + \boldsymbol{\phi}_{2i+1,2j+1}\right), && 0 \le i < H^{(1)}, 0 \le j < W^{(1)} \\ D_{ij}^{(2)}(\boldsymbol{\phi}) &= \boldsymbol{\phi}_{i_0+i,j_0+j}, && 0 \le i < H^{(2)}, 0 \le j < W^{(2)} \end{aligned} \tag{D.5}$$

其中 i_0 和 j_0 是 $\boldsymbol{\phi}^{(2)}$ 相对于 $\boldsymbol{\phi}$ 的偏移量（索引从 0 开始）。

总重建损失定义为各个 AE_k的重建损失的加权和：

$$\ell(\tilde{\boldsymbol{\phi}}, \boldsymbol{\phi}) = w^{(1)}\ell^{(1)}(\tilde{\boldsymbol{\phi}}^{(1)}, \boldsymbol{\phi}^{(1)}) + w^{(2)}\ell^{(2)}(\tilde{\boldsymbol{\phi}}^{(2)}, \boldsymbol{\phi}^{(2)}), \tag{D.6}$$

其中 $w^{(k)}$ 是 权重因子。由于各个 AE_ks 相互独立，权重 $w^{(k)}$ 实际上决定了它们之间的比率。为简单起见，我们取 $w^{(1)} = w^{(2)} = 1$。

相对学习

单项损失 $l^{(k)}$ 考虑到，我们不希望浪费 AE_1 有限的表达能力，去重建 AE_2 已经重点关注的部分。此外，为了减轻边界效应（在边界处，重建效果可能较差），我们还希望将边缘部分排除在 AE_2 的重构损失之外。因此，我们采用加权（相对）重建损失：

$$\ell^{(k)}\left(\tilde{\boldsymbol{\phi}}^{(k)},\boldsymbol{\phi}^{(k)}\right)=\frac{\sum_{ij}\alpha_{ij}^{(k)}\left(\tilde{\boldsymbol{\phi}}_{ij}^{(k)}-\boldsymbol{\phi}_{ij}^{(k)}\right)^2}{\sum_{ij}\alpha_{ij}^{(k)}\left(\boldsymbol{\phi}_{ij}^{(k)}\right)^2+W^{(k)}H^{(k)}\epsilon_\phi}$$

对于相对均方误差损失，或

$$\ell^{(k)}\left(\tilde{\boldsymbol{\phi}}^{(k)},\boldsymbol{\phi}^{(k)}\right)=\frac{\sum_{ij}\alpha_{ij}^{(k)}\left|\tilde{\boldsymbol{\phi}}_{ij}^{(k)}-\boldsymbol{\phi}_{ij}^{(k)}\right|}{\sum_{ij}\alpha_{ij}^{(k)}\left|\boldsymbol{\phi}_{ij}^{(k)}\right|+W^{(k)}H^{(k)}\epsilon_\phi}$$

对于相对 L_1 损失。这里，场 $\tilde{\boldsymbol{\phi}}^{(k)}$ 表示自编码器重构，$\boldsymbol{\phi}^{(k)}$ 表示输入和目标，$\epsilon_\phi>0$ 表示用于防止发散梯度的归一化偏移量。权重因子 $\boldsymbol{\alpha}^{(k)}$ 经过选择，使圆柱中心不会影响 AE_1 的损失，且第二子域边缘不会影响 AE_2的损失：

$$\begin{aligned}\boldsymbol{\alpha}^{(1)}&=1-\mathbf{D}^{(1)}(\mathbf{S}(d^{(1)})),\\ \boldsymbol{\alpha}^{(2)}&=\mathbf{D}^{(2)}(\mathbf{S}(d^{(2)})),\end{aligned}\tag{D.7}$$

其中 $\mathbf{S}$：$\mathbb{R}\to\mathbb{R}^{H\times W}$ 是一个二维平滑矩形函数：

$$\begin{aligned}S_{ij}(d)&=S\left(\frac{\min\{i'-i_0,i_1-i'\}-d}{s}\right)\\&\quad\times S\left(\frac{\min\{j'-j_0,j_1-j'\}-d}{s}\right),\\ i'&=i+\frac{1}{2},\qquad\text{(for cell-centered values)}\\ j'&=j+\frac{1}{2},\\ S(x)&=\frac{1}{1+e^{-x}},\quad\text{(sigmoid)}\end{aligned}\tag{D.8}$$

其中 (i_0,j_0) 和 $(i_1,j_1)=(i_0+H^{(2)},j_0+W^{(2)})$ 分别是 AE_2的起始和结束边界。参数 $s>0$是平滑因子，$d^{(1)},d^{(2)}>0$ 是空间边距。

最终的 $\boldsymbol{\phi}\in\mathbb{R}^{H\times W}$ 重构也以加权方式由 $\boldsymbol{\phi}^{(1)}$ 和 $\boldsymbol{\phi}^{(2)}$ 完成：

$$\begin{aligned}\boldsymbol{\phi}&=\boldsymbol{\beta}^{(1)}\odot\mathbf{U}(\boldsymbol{\phi}^{(1)})+\boldsymbol{\beta}^{(2)}\odot\mathbf{U}(\boldsymbol{\phi}^{(2)}),\\ \boldsymbol{\beta}^{(1)}&=1-\mathbf{S}(d^r)\\ \boldsymbol{\beta}^{(2)}&=\mathbf{S}(d^r)\end{aligned}\tag{D.9}$$

其中，$\mathbf{U}^{(k)}$：$\mathbb{R}^{H^{(k)}\times W^{(k)}}\to\mathbb{R}^{H\times W}$ 表示上采样操作，d^r 表示重构边距，而算子$\odot$ 表示逐元素乘法。在本案例研究中，$\mathbf{U}^{(1)}$ 表示采用双线性插值的上采样操作，而$\mathbf{U}^{(2)}$ 表示零填充操作。

我们测试了 AE_2 的两个变体，它们以不同的分辨率运行：256×256 和 224×224。如第 5.1.3 节所示，后一个变体在宏利用率较高时表现出略优的性能，因此被选入参考参数集。参数见表 D.1，几何形状如图 D.1 所示。通过采用边距 $d^{(1)}>d^r>d^{(2)}$，我们确保 AE_1 和 AE_2 都能准确重建发生平滑混合的部分（$0<\beta_{ij}^{(k)}<1$）。

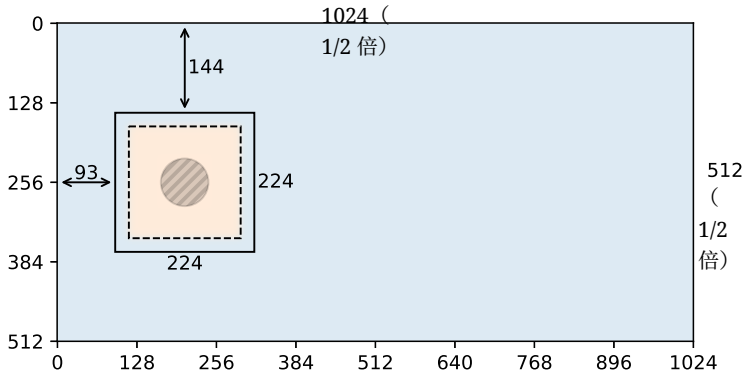

图 D.1：分辨率为 224×224 的多分辨率自编码器的几何形状，其中 AE_2。实线内框表示 AE_2 所作用的子域，虚线表示距内部子域边界 $d^r = 22$ 的距离（其中 $\boldsymbol{\beta}^{(1)} = \boldsymbol{\beta}^{(2)} = 0.5$）。颜色表示混合贡献（$\boldsymbol{\beta}^{(1)} > 0.5$ 标记为蓝色，$\boldsymbol{\beta}^{(2)} > 0.5$ 为橙色），圆形表示圆柱体。

表 D.1：几何形状 以及多分辨率自编码器的损失函数参数，还有每个单独$\mathbf{u}$ 场的内存使用量
单精度 ision.

参数	**变体 1**	变体 2
原始分辨率	1024×512	
AE_1 分辨率	512×256	
AE_2 分辨率	256×256	$\mathbf{224 \times 224}$
AE_2 开始	(77, 128)	(93, 144)
AE_2 结束	(333, 384)	(317, 368)
权重 $w^{(1)}$	1.0	
权重 $w^{(2)}$	1.0	
边距 $d^{(1)}$	30.0	
边距 d^r	22.0	
边距 $d^{(2)}$	14.0	
平滑 s	3.0	
原始大小为$\mathbf{u}$	4.00 MB	
压缩后大小为 $\mathbf{u}$	1.50 MB	1。38 MB

多分辨率方法将内存和存储需求降低了 2.9 倍（对于 224×224 变体），并使训练速度提高了约相同的倍数，但准确度会略有下降。具体而言，原始速度场 $\mathbf{u}$ 与下采样–上采样后的 $\mathbf{u}'$ 之间的相对均方误差为 $\|\mathbf{u}' - \mathbf{u}\|_2^2 / \|\mathbf{u}\|_2^2 \approx 10^{-6}$。

D.3. 自编码器摘要与损失函数

以上讨论总结如下。状态 $\mathbf{u}$ 在数据集中以两个下采样版本 $\mathbf{u}^{(1)}$ 和 $\mathbf{u}^{(2)}$存储，各自由一个编码器—解码器对 AE_k处理。对于每个 AE_k,$k \in \{1, 2\}$，编码器 k 以经过下采样的速度场 $\mathbf{u}^{(k)} \in \mathbb{R}^{H^{(k)} \times W^{(k)} \times 2}$ 为输入，并将其压缩为潜在状态 $\mathbf{z}^{(k)} \in \mathbb{R}^{d_z^{(k)}}$。解码器 k 接收潜在状态 $\mathbf{z}^{(k)}$，并将其解压缩为标量场 $\tilde{\psi}^{(k)}$（每个维度的两侧各填充一个单元格）。随后，根据式 (D.3) 从 $\tilde{\psi}^{(k)}$ 计算重构速度 $\tilde{\mathbf{u}}^{(k)}$。AE_k 的重构损失定义为 $\mathbf{u}^{(k)}$ 的相对均方误差损失与涡量的相对 L_1 损失 $\boldsymbol{\omega}^{(k)}$ 的加权和（为简洁起见，以下省略记号 (k)）

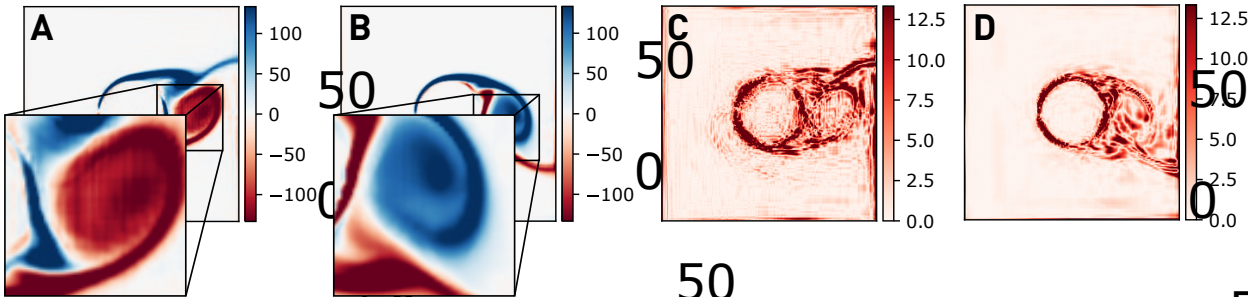

图 D.2：AE_2, 在 $\lambda_\omega = 0$ （不含涡量损失，A 和 C）和 $\lambda_\omega = 0.03$ （含涡量损失，B 和 D）条件下的重构涡量场（A 和 B）及其绝对误差（C 和 D）。

为简洁起见）：

$$\ell^{(k)}(\tilde{\mathbf{u}}, \mathbf{u}) = \lambda_u \frac{\sum_{ij} \alpha_{ij} (\tilde{\mathbf{u}}_{ij} - \mathbf{u}_{ij})^2}{\sum_{ij} \alpha_{ij} \mathbf{u}_{ij}^2 + W^{(k)} H^{(k)} \epsilon_u} + \lambda_\omega \frac{\sum_{ij} \alpha'_{ij} \|\tilde{\boldsymbol{\omega}}_{ij} - \boldsymbol{\omega}_{ij}\|_1}{\sum_{ij} \alpha'_{ij} \|\boldsymbol{\omega}_{ij}\|_1 + (W^{(k)} - 2)(H^{(k)} - 2)\epsilon_\omega}, \tag{D.10}$$

$$\tilde{\boldsymbol{\omega}} = (\nabla \times \tilde{\mathbf{u}})_z, \quad \boldsymbol{\omega} = (\nabla \times \mathbf{u})_z,$$

$$\boldsymbol{\alpha}' = \boldsymbol{\alpha}_{1..H-2;1..W-2} \in \mathbb{R}^{(H-2)\times(W-2)}, \tag{D.11}$$

其中， $\lambda_u = 1$ 和 $\lambda_\omega = 0.03$ 是权重因子， $\epsilon_u = 0.01$ 和 $\epsilon_\omega = 0.7$ 是归一化偏移量，用于避免在初始状态满足 $\mathbf{u} \approx \mathbf{0}$ 时进行训练时出现梯度爆炸。 λ_ω 的影响如图 D.2 所示。选择 ϵ_u 和 ϵ_ω ，使其分别与 $\mathrm{Re} = 500$ 下已形成流动的均值（ $\mathbf{u}_{ij}^{(1)}$ ）的 $\approx 25\%$ 以及均值 $\left|\omega_{ij}^{(1)}\right|$ 相匹配。总损失 ℓ （...）定义为 AE_k 各项损失的加权和：

$$\ell(\tilde{\mathbf{u}}^{(1)}, \tilde{\mathbf{u}}^{(2)}, \mathbf{u}^{(1)}, \mathbf{u}^{(2)}) = w^{(1)} \ell^{(1)}(\tilde{\mathbf{u}}^{(1)}, \mathbf{u}^{(1)}) + w^{(2)} \ell^{(2)}(\tilde{\mathbf{u}}^{(2)}, \mathbf{u}^{(2)}), \tag{D.12}$$

其中 $w^{(1)} = w^{(2)} = 1$ 是 AE_k 之间的相对权重。无论是在计算式（13）中的在线验证误差E 时，还是在执行宏观到微观转换时，完整分辨率的速度$\mathbf{u}$ 都是通过按式（D.9）所述合并$\mathbf{u}^{(1)}$ 和$\mathbf{u}^{(2)}$ 重建的。合并必须针对速度$\mathbf{u}$ 进行，而不能针对流函数 ψ 进行。这是因为流函数的定义存在一个未指定的加性常数，因此无法将它们合并。此外，通过对两个上采样后的速度场进行平滑混合（式（D.9））， 我们确保$\mathbf{u}$ 的空间导数平滑。

D.4. 超参数、*CNN*架构和训练

除了第 5.1.2 节自编码器研究中列出的超参数外，批量大小等其他参数也经过了手动调优，并列于表 D.2 中。

$\mathbf{u}$自编码器的基础是两个卷积自编码器，它们分别处理一个下采样数组$\mathbf{u}^{(k)}$。这两个自编码器采用相同的架构，但分别进行训练。经过超参数研究后得到的最终 CNN架构见表 D.3。

训练与模拟和 AdaLED 推理同步持续进行。在每个周期中，网络使用数据集的 12.5% 进行训练。一个周期包括训练自编码器、对状态进行编码以构建长短期记忆网络的临时数据集，最后训练长短期记忆网络。三个训练阶段的相对执行时间如图D.3所示。

D.5. 潜在轨迹

图D.4展示了第 5.1.1 节模拟中的宏观轨迹的一部分。前16行对应潜在状态 $\mathbf{z}(t)$，最后两行对应力 $\mathbf{F}_{\mathrm{cyl}}(t)$（缩放因子为

表 D.2：圆柱体尾流研究中手动调整的网络超参数。其他参数列于第 5.1.3 节。

超参数	值
自编码器批量大小	8
LSTM批量大小	8
LSTM隐藏状态大小	32
LSTM层数	2
单精度与双精度	单一

表 D.3：圆柱后方流动案例研究中卷积自编码器的架构。所有卷积层均采用填充_模式=复制填充。

ID	自编码器 #1 形状	自编码器 #2 形状	层
	$2 \times 256 \times 512$	$2 \times 224 \times 224$	输入
1	$16 \times 256 \times 512$	$16 \times 224 \times 224$	卷积(2, 16, 核_大小=5, 填充=2)
2	$16 \times 128 \times 256$	$16 \times 112 \times 112$	平均池化(kernel_size=2, stride=2)
3	$16 \times 128 \times 256$	$16 \times 112 \times 112$	CELU()
4	$16 \times 128 \times 256$	$16 \times 112 \times 112$	卷积(16, 16, kernel_size=5, 填充=2)
5	$16 \times 64 \times 128$	$16 \times 56 \times 56$	平均池化(kernel_size=2, stride=2)
6	$16 \times 64 \times 128$	$16 \times 56 \times 56$	CELU()
7	$16 \times 64 \times 128$	$16 \times 56 \times 56$	卷积(16, 16, kernel_size=5, 填充=2)
8	$16 \times 32 \times 64$	$16 \times 28 \times 28$	平均池化(kernel_size=2, stride=2)
9	$16 \times 32 \times 64$	$16 \times 28 \times 28$	CELU()
13	$16 \times 32 \times 64$	$16 \times 28 \times 28$	卷积(16, 16, kernel_size=3, 填充=1)
14	$16 \times 16 \times 32$	$16 \times 14 \times 14$	平均池化(kernel_size=2, stride=2)
15	$16 \times 16 \times 32$	$16 \times 14 \times 14$	CELU()
16	8192	3136	展平()
17	$d_z^{(1)} = 8$	$d_z^{(2)} = 8$	线性层()
18	$d_z^{(1)} = 8$	$d_z^{(2)} = 8$	双曲正切函数()
	$d_z^{(1)} = 8$	$d_z^{(2)} = 8$	$\mathbf{z}^{(i)}$
1	8192	3136	线性层()
2	$16 \times 16 \times 32$	$16 \times 14 \times 14$	视图层()
3	$16 \times 32 \times 64$	$16 \times 28 \times 28$	上采样(scale_factor=2.0, mode=双线性))
4	$16 \times 32 \times 64$	$16 \times 28 \times 28$	卷积(16, 16, kernel_size=3, 填充=1)
8	$16 \times 32 \times 64$	$16 \times 28 \times 28$	CELU()
9	$16 \times 64 \times 128$	$16 \times 56 \times 56$	上采样(scale_factor=2.0, mode=双线性))
10	$16 \times 64 \times 128$	$16 \times 56 \times 56$	卷积(16, 16, kernel_size=5, 填充=2)
11	$16 \times 64 \times 128$	$16 \times 56 \times 56$	CELU()
12	$16 \times 128 \times 256$	$16 \times 112 \times 112$	上采样(scale_factor=2.0, mode=双线性))
13	$16 \times 128 \times 256$	$16 \times 112 \times 112$	卷积(16, 16, 卷积核_大小=5, 填充=2)
14	$16 \times 128 \times 256$	$16 \times 112 \times 112$	CELU()
15	$16 \times 256 \times 512$	$16 \times 224 \times 224$	上采样(scale_factor=2.0, mode=双线性))
16	$1 \times 258 \times 514$	$1 \times 224 \times 224$	卷积(16, 1, kernel _size=5, 填充=3)
17	$2 \times 256 \times 512$	$2 \times 224 \times 224$	流函数到速度() (式 (D.3))
	171K	85K	参数总数

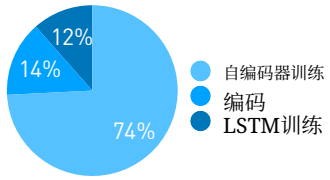

图 D.3：圆柱后方流动案例中各训练阶段执行时间所占的比例。

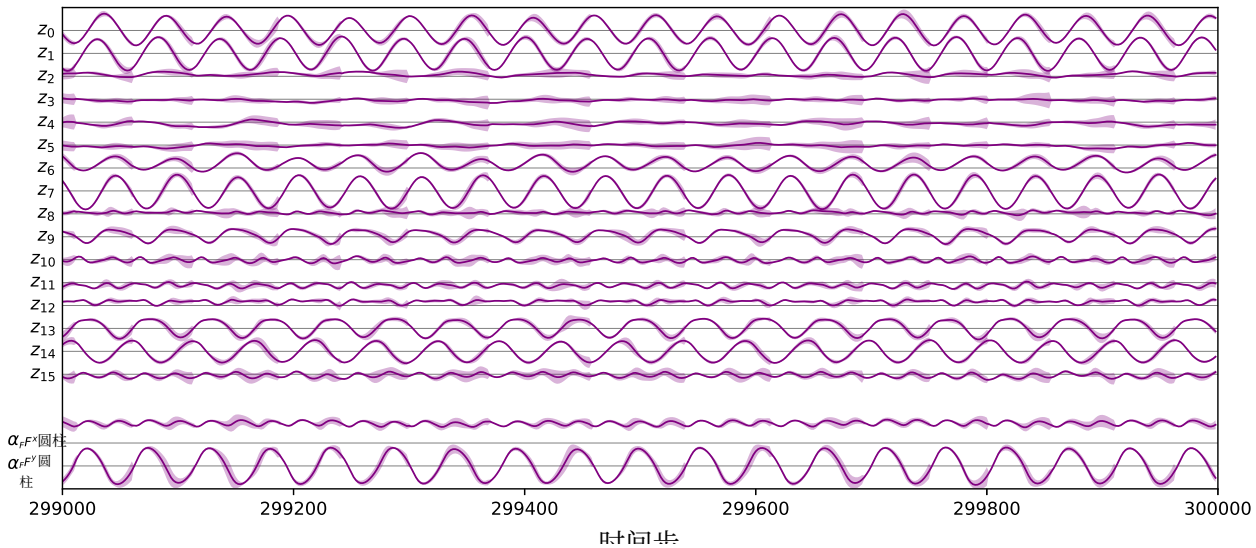

图 D.4：集成预测了第 5.1.1 节模拟中的轨迹 $\mathbf{z}(t)$ 和 $\alpha_F \mathbf{F}^{\text{cyl}}(t)$。实线表示集成平均预测，浅色区域表示预测不确定性（集成的标准差）。为清晰起见，不确定性放大了 8 倍。数值范围约为 -0.5 至 0.5。参见附录 D.5。

$\alpha_F = 7.2$)。总不确定性 σ 定义为全部 18 个不确定性的均方根。可以看出，大部分不确定性来自低振幅潜变量。根据各变量的重要性对不确定性加权，是未来研究的一个方向。

D.6. 推广至其他雷诺数剖面

第 5.1.1 节所报告模拟中使用的超参数和阈值，针对在雷诺数 $= 600$、750 和 900 之间循环变化、每 5000 个时间步更新一次的特定雷诺数剖面进行了微调，具体说明见第 5.1.3 节（超参数研究使用的模拟时长短于生产运行）。为检验超参数和阈值能否推广到另一种雷诺数剖面，我们模拟了雷诺数在 500 和 1000 之间交替变化、每 10000 个时间步更新一次的情况。图 D.5 显示了宏观利用率 η、速度场误差 E和圆柱体受力误差 E_F。与第 5.1.1 节中 $\eta = 69\%$ 的宏观利用率（加速比为 2.9 倍）相比，此处达到的利用率为 58%（加速比为 2.1 倍）。与之前一样，平均速度误差和圆柱体受力误差 E 和 E_F 分别为 1% 和 5%。在这种情况下，更改设置导致加速比较低。因此，要达到最佳性能，如果更新了仿真设置，可能需要进一步微调超参数（尤其是学习率和阈值）。

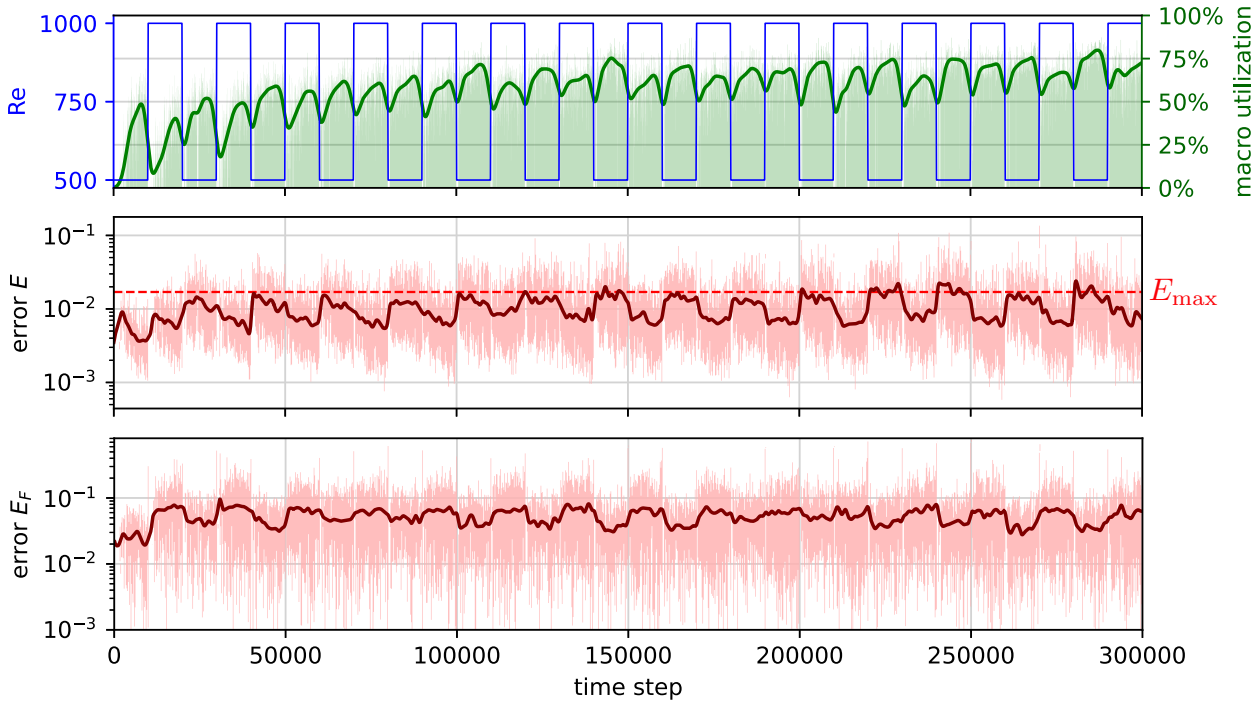

图 D.5：在雷诺数 $\mathrm{Re}(t) \in \{500, 1000\}$ 条件下，AdaLED 在圆柱体后方流动模拟中的性能，与图 9 类似。上：雷诺数 $\mathrm{Re}(t)$ 剖面和宏观利用率 η。中部和下部：速度的验证误差（E，式 (13)）和圆柱体上的力的验证误差（E_F，式 (14)）。每步误差（淡红色）在仅宏观阶段开始时的较低值和该阶段结束时的较高误差之间交替变化。