arXiv:2302.01278v2 [math.DS] 3 Feb 2023

用于纳维–斯托克斯方程低维参数化的卷积自编码器、聚类与 POD

Yongho Kim*, † Jan Heiland*, †

*Max Planck Institute for Dynamics of Complex Technical Systems, Magdeburg, Germany
†Department of Mathematics, Otto von Guericke University Magdeburg, Magdeburg, Germany电子邮箱：ykim@mpi-magdeburg.mpg.de，ORCID：0000-0003-4181-7968电子邮箱：heiland@mpi-magdeburg.mpg.de，ORCID：0000-0003-0228-8522

摘要： 大规模动力系统的仿真需要耗费大量计算资源。高维状态的低维参数化（例如本征正交分解（POD））可以在精度与模型复杂度之间提供一定折衷，从而减轻计算负担。然而，对于真正的低维参数化（例如用于控制器设计），POD 等线性方法已达到其固有极限，因此非线性方法将成为首选。本文提出一种由非线性编码器和仿射线性解码器构成的卷积自编码器（CAE），并研究其与 k 均值聚类的组合，以提升编码性能。我们在两个由不可压缩纳维–斯托克斯方程建模的圆柱绕流场景中，将所提出的方法与标准 POD 方法进行了比较。

关键词： 卷积自编码器、聚类、线性参数变化（LPV）系统、模型降阶、不可压缩流。

AMS 学科分类： MSC1,MSC2, MSC3

创新性声明：

- 用于低维参数化有限元数据的卷积自编码器（CAE）。
- 将编码器和解码器与聚类技术相结合，以提升性能。
- 在两个圆柱体尾流仿真场景中，全面展示卷积自编码器结合聚类的能力，并与 POD 进行比较。

1 引言

准确的有限元方法 (FEM)流体流动离散化很容易产生具有数百万个自由度的动力系统。这给仿真带来了计算挑战；由于涉及非线性，基于这些模型进行计算机辅助控制器设计更是如此。

模型降阶技术的核心思想与潜力，在于识别并利用低维坐标，以良好地表示或近似通常具有高维度的系统的动力学。例如，投影方法 例如广受认可的本征正交分解 (POD)基于向子空间进行线性投影，这些子空间经过设计，能够编码系统中最相关的动力学。这些方法可以高效降低给定动力系统状态空间的维度，从而显著减少仿真所需的内存和计算资源。然而，对于给定数量的自由度，线性方法的精度自然受到限制，这一点可由*Kolmogorov n-*宽度加以表述，参见例如[20]。

因此，如果希望在保持令人满意的精度的同时实现极低维参数化，可能需要采用非线性方法；参见 [12, 18, 21]。

因此，鉴于控制器设计更看重低维度而非实际精度，我们考虑通过可能的非线性映射，在低维流形上近似并参数化状态。

参数化是指用不同的坐标表示给定系统的状态；近似则意味着设计这种表示，以牺牲精度为代价，使参数化坐标的维度尽可能低。编码 是指对状态的降维参数化表示进行计算，而解码 或重构 则描述了从参数化表示计算实际坐标下状态的过程。相应的机制或算法称为编码器 或解码器。

因此，POD 方法可以看作一种基于线性投影的编码器/解码器。如上所述，线性方法存在局限，因此，诸如通用自编码器（AE）之类的非线性方法逐渐兴起，作为 POD 的替代方案或增强手段 [8]。

自编码器 （例如参见 [11, 第 14]章）通常指一种神经网络，旨在高效地对数据进行编码，并且仅通过考察数据本身来训练。其网络架构可针对具体问题设计，并包含多个隐藏层。编码和解码中的非线性可能来自激活函数的使用。基于这些原理，人们开发了许多类型的自编码器，例如稀疏自编码器、去噪自编码器、收缩自编码器和卷积自编码器。（例如参见 [3]）

十多年来，随着用于设计和训练神经网络的计算设施与工具箱日益普及，自编码器一直被视为模型降阶领域中的一种替代方案。一个近期的例子采用改进的非线性自编码器来近似线性化双连杆平面机器人操纵器的系数矩阵元素；参见[16]。其他例子则通过深度卷积自编码器求解对流主导问题；例如参见 [18]）

以偏微分方程作为底层模型时，基于所谓的卷积神经网络（CNN）的自编码器变体似乎尤其有用，这一点可从多项使用卷积神经网络求解偏微分方程的尝试中看出；当前的相关研究示例见 [9, 14, 24]。将变量从一层传播到另一层的线性部分实现为卷积，可通过强制采用稀疏连接大幅减少网络的参数，这意味着在各层中，每个变量只与有限数量的其他变量相关。此外，配套的池化操作将若干相邻变量合并为一个，从而降低每一层的变量维数。在经典的有限元法中，稀疏连接是衡量性能的一项重要标准，通常通过采用具有局部支撑的基函数来实现，而池化则可与多尺度方法联系起来。

一旦获得状态的极低维编码，解码器的任务就是根据该编码重构高维状态。直观地说，输出空间中需要重现的变化越多，解码器的设计就越复杂。因此，我们认为，在降阶坐标中进行聚类能够将状态空间有效分离，使得各个独立的解码器只需覆盖整体变化范围中的某一部分；实现示例见 [13]。

聚类 算法用于区分数据，并在不使用标签的情况下将给定数据集划分为若干子集，也就是说，无需预先了解子集的特征；例如，参见

[22] 关于聚类的一般介绍，以及 [23] 对最流行的聚类算法——k-均值聚类 ——的讨论。该算法基于距离度量，迭代地识别数据集中预先设定数量的簇。此外，聚类算法与神经网络能够很好地结合；而且，在数据维度较低时，例如在低维特征空间中，聚类会变得更容易；例如，参见[2, 7, 19]。

由于聚类不是平滑操作，直到最近，这种方法才开始应用于动力系统。尽管如此，人们已经发现，采用聚类的降维技术能够通过分别处理大规模数据集的子集来提升检索性能并降低计算成本，例如在文本检索系统中；参见 [10]。

本文着重于识别状态的极低维参数化（而非低维动力学模型），因此非光滑性并非问题，我们可以在数值研究中直接采用聚类。

本文结构安排如下：在第 2 节中，我们介绍不可压缩纳维-斯托克斯方程，该方程将作为本文研究的有限元法模型。在第 3 节中，我们定义低维参数化方法。在第 4 节中，我们分别讨论如何利用所考虑的方法构建各个降阶模型。在第 5 节中，我们展示各方法的重构误差结果，并根据残差评估降阶模型。在第 6 节中，我们总结全文，讨论局限性，并提出未来工作的潜在研究方向。

2 不可压缩纳维–斯托克斯方程

我们考虑　空间离散化的不可压缩纳维–斯托克斯方程动力系统

$$\mathbf{M}\dot{\mathbf{v}}(t) + \mathbf{N}(\mathbf{v}(t))\mathbf{v}(t) + \mathbf{A}\mathbf{v}(t) - \mathbf{J}^\top \mathbf{p}(t) = \mathbf{f}(t) \tag{1a}$$

$$\mathbf{J}\mathbf{v}(t) = 0, \tag{1b}$$

其中，对于时间 $t > 0$，$\mathbf{v}(t) \in \mathbb{R}^{n_v}$ 和 $\mathbf{p}(t) \in \mathbb{R}^{n_p}$ 分别表示有限元网格上速度和压力的状态，并且 $\mathbf{M} \in \mathbb{R}^{n_v \times n_v}$ 是质量矩阵，其中 $\mathbf{A} \in \mathbb{R}^{n_v \times n_v}$ 对扩散进行建模，其中 $\mathbf{N}\colon \mathbb{R}^{n_v} \to \mathbb{R}^{n_v \times n_v}$ 表示离散对流项，并且 $\mathbf{J} \in \mathbb{R}^{n_p \times n_v}$和 $\mathbf{J}^\top$ 分别表示离散散度和梯度；有关技术细节和示例离散化，请参见 [4] 。

下文中，我们将省略变量 $\mathbf{v}$ 的时间依赖关系（之后也会省略 $\boldsymbol{\rho}$ 和 $\tilde{\mathbf{v}}$ 的时间依赖关系，后者将表示编码和重构）。

正如至少在理论考量中通常所采用的那样，系统（1）可以表示为常微分方程。在此，我们将采用常微分方程表述，以避免在数值格式中处理 p 变量时遇到的技术困难（参见 [1]），并使其能够直接适用于其他常微分方程模型。

我们简要说明如何得到不可压缩纳维–斯托克斯方程的常微分方程表述。假设 $\mathbf{J}$ 满秩，且 $\mathbf{M}$ 可逆且对称正定，这是一个合理的假设。利用方程 $\mathbf{p} = \mathbf{S}^{-1}\mathbf{J}\mathbf{M}^{-1}(\mathbf{N}(\mathbf{v})\mathbf{v} + \mathbf{A}\mathbf{v} - \mathbf{f})$，其中 $\mathbf{S} = \mathbf{J}\mathbf{M}^{-1}\mathbf{J}^\top$，我们可以消去 $\mathbf{p}$，由（1）得到

$$\mathbf{M}\dot{\mathbf{v}} + (\mathbf{I} - \mathbf{J}^\top \mathbf{S}^{-1}\mathbf{J}\mathbf{M}^{-1})(\mathbf{N}(\mathbf{v})\mathbf{v} + \mathbf{A}\mathbf{v} - \mathbf{f}) = 0.$$

利用投影算子 $\Pi := \mathbf{I} - \mathbf{M}^{-1}\mathbf{J}^\top \mathbf{S}^{-1}\mathbf{J}$，并确认解 v 满足 $v = \Pi v$后，我们可以指出，在计算 v时，系统（1）可以替换为

$$\mathbf{M}\dot{\mathbf{v}} + \Pi^\top(\mathbf{N}(\mathbf{v})\mathbf{v} + \mathbf{A}\mathbf{v} - \mathbf{f}) = 0. \tag{2}$$

在接下来的章节中，我们将通过多种指标考察状态$\mathbf{v}$ 能否被编码为非常低维的编码 ρ。首先，我们考察标准重构误差

$$\|\mathbf{v}(t) - \tilde{\mathbf{v}}(t)\|_{\mathbf{M}}$$

该误差是在所谓的$\mathbf{M}$-范数下测量的。此范数由质量矩阵$\mathbf{M}$ 诱导，是与底层模型问题相关的 L^2 *Sobolev* 范数的离散且一致的对应形式。其次，我们通过测量来考察相对于由此产生的对流的重构

$$\|\mathbf{N}(\tilde{\mathbf{v}}(t))\mathbf{v}(t) - \mathbf{N}(\mathbf{v}(t))\mathbf{v}(t)\|_{\mathbf{M}^{-1}},$$

其中，选择$\mathbf{M}^{-1}$ 范数，以适应有限元法的背景；在该背景下，$\mathbf{N}(\mathbf{v}(t))\mathbf{v}(t)$是一个泛函。最后，考虑到可能需要构建相关的低阶动力学代理模型，我们将报告数据点处的残差。

3 低维参数化

本节将介绍若干可能为非线性的方法，用于编码和解码，重点关注极低维参数化，并以标准方法 POD 作为基准。

如上所述，我们用$\mathbf{V}$ 表示取值于 $\mathbb{R}^{n_v}$的速度，用 $\boldsymbol{\rho}$ 表示取值于 $\mathbb{R}^{n_\rho}$的编码，并用 $\tilde{}\mathbf{v}$ 表示由编码重构得到的速度。考虑的方法如下：

3.1 本征正交分解（POD）

POD 方法 [5] 很容易被理解为一个自编码器，其包含

(1) 一个线性编码器，定义为

$$\boldsymbol{\rho} = \mathbf{V}^\top \mathbf{v}$$

(2) 以及一个线性解码器

$$\tilde{\mathbf{v}} = \mathbf{V}\boldsymbol{\rho},$$

其中，$\mathbf{V} \in \mathbb{R}^{n_v \times n_\rho}$ 是一个 POD基，通常通过对变量 $\mathbf{v}$ 的快照矩阵求取其 n_ρ 领先奇异向量来计算得到。

3.2 卷积神经网络（CNN）

为了应用标准卷积方法，输入数据需要位于张量网格上，也就是说，网格由矩形构成，不含悬挂节点，且与坐标轴对齐；示意见 [13, 图 3， 4] 展示了这种插值。为此，使用相应的插值矩阵 $\mathbf{I}_c$ ，使得 CNN 输入 $\mathbf{v}_{\mathsf{CNN}}(t)$ 可通过线性变换 $\mathbf{v}_{\mathsf{CNN}}(t) = \mathbf{I}_c\mathbf{v}(t)$ 生成。我们考虑 [12]中介绍的卷积神经网络架构，该架构利用本征正交分解模态进行重构。因此，卷积神经网络被定义为由以下部分组成的模型：

(1) 一个 非线性卷积编码器 μ:

$$\boldsymbol{\rho} = \mu(\mathbf{v}_{\mathsf{CNN}})$$

(2) 和一个线性解码器

$$\tilde{\mathbf{v}} = \phi(\boldsymbol{\rho}) = \mathbf{V}(\mathbf{W}\boldsymbol{\rho}),$$

其中$\mathbf{v}_{\mathsf{CNN}} \in \mathbb{R}^{2\times w\times h}$ 是卷积神经网络的输入，$\mathbf{V} \in \mathbb{R}^{n_v \times r}$ 是具有 r 个模态的 POD基，$\mathbf{W} \in \mathbb{R}^{r\times n_\rho}$ 是一个可训练矩阵。式（1）中解码器的前馈网络 μ 具体描述如下：

$$\begin{aligned}
&\mathbf{u}^{(m)} = \mathbf{a}(\mathbf{F}^{(m)}(\mathbf{u}^{(m-1)})),\ \mathbf{u}^{(0)} = \mathbf{v}_{\mathsf{CNN}}, \quad m = 1, \cdots, M-1,\\
&\mathbf{u}^{(M)} = \mathbf{F}^{(M)}(\mathbf{u}^{(M-1)}),\\
&\boldsymbol{\rho} = \mathbf{U}\bar{\mathbf{u}}^{(M)} + \boldsymbol{\beta}, \quad \mathbf{U} \in \mathbb{R}^{n_\rho \times n_u},\ \boldsymbol{\beta} \in \mathbb{R}^{n_\rho},
\end{aligned}$$

其中，$\mathbf{a}$ 是非线性激活函数， M 是卷积层的数量，$\mathbf{F}$ 是卷积层， $\bar{}\mathbf{u}^{(M)}$ 是向量化后的$\mathbf{u}^{(M)}$， n_u 是 $\bar{}\mathbf{u}^{(M)}$ 的维度。

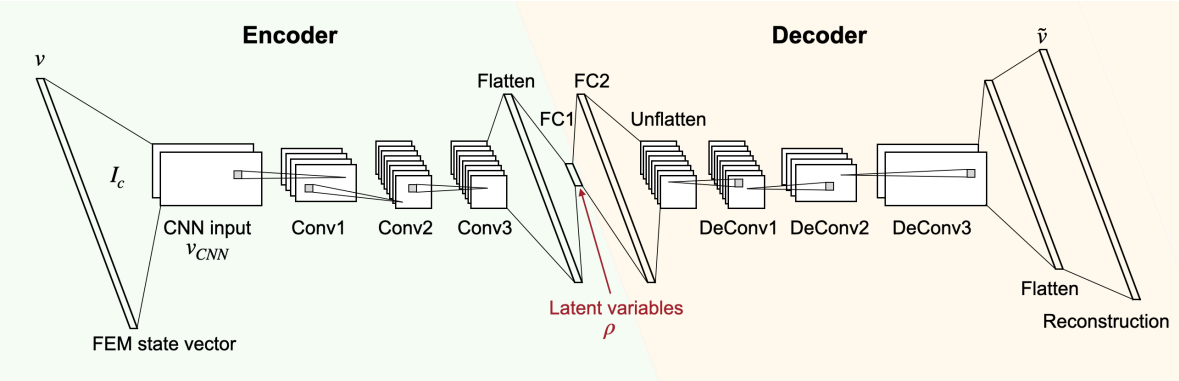

图 1：用于重构有限元法状态向量的卷积自编码器

3.3 卷积自编码器（CAE）

除了使用本征正交分解模态进行重构外，我们还提出使用所谓的转置卷积层，不设置非线性激活，并配合插值算子$\mathbf{I}_P$，将数值从张量化网格插值回有限元网格。这样，我们就可以研究卷积神经网络在重构方面的特性，同时将整个过程保持为仿射线性——因为它是仿射线性算子的复合。

因此，该模型由以下部分组成：

(1) a非线性卷积编码器

$$\mu\colon \mathbf{v}_{\mathsf{CNN}} \to \boldsymbol{\rho}$$

and

(2) an仿射线性反卷积解码器

$$\tilde{\mathbf{v}} = \phi(\boldsymbol{\rho}) = \mathbf{D}\boldsymbol{\rho} + \mathbf{b}$$

其中，$\mathbf{D} \in \mathbb{R}^{n_v \times n_\rho}$ 是一个矩阵，而$\mathbf{b} \in \mathbb{R}^{n_v}$ 是一个向量。

在实践中，解码器 $\phi\colon \boldsymbol{\rho} \to \tilde{}\mathbf{v}$ 通过以下方式实现：

$$\begin{aligned}
&\bar{\mathbf{z}}^{(0)} = \mathbf{U}\boldsymbol{\rho} + \boldsymbol{\beta}, \quad U \in \mathbb{R}^{n_u \times n_\rho}, \boldsymbol{\beta} \in \mathbb{R}^{n_u}, \\
&\mathbf{z}^{(m)} = \mathbf{G}^{(m)}(\mathbf{z}^{(m-1)}),\ m = 1, \cdots, M, \quad \text{with } \mathbf{z}^{(0)} \text{ as the unflattened } \bar{\mathbf{z}}^{(0)} \\
&\tilde{\mathbf{v}} = \mathbf{I}_p \bar{\mathbf{z}}^{(M)}, \quad \mathbf{I}_p \in \mathbb{R}^{n_z \times n_v},
\end{aligned}$$

其中，M 是层数，$\mathbf{G}$ 是一个反卷积层，$\bar{}\mathbf{z}^{(M)}$ 是向量化后的$\mathbf{z}^{(M)}$，而 n_z 是 $\bar{}\mathbf{z}^{(M)}$的维数。该架构如图 1 所示。

3.4 聚类本征正交分解（cPOD）

正如引言中所述，根据不同的簇使用多个解码器可以有效提升重构性能。在使用通用自编码器进行聚类之前，我们将聚类本征正交分解 (cPOD)定义为一种由以下部分组成的模型

(1) 一个线性编码器

$$\boldsymbol{\rho} = \mathbf{V}^\top \mathbf{v},$$

表 1：基于卷积神经网络的模型描述

–	CNN	CAE
#编码层	4 + 全连接层	3 + 全连接层
#卷积通道	(4，8，10，12)	(4,8,8)
#解码层	fc +V	fc +3+ fc
#反卷积通道	-	(8，8，4)

* fc：全连接层，$\mathbf{V}$：包含 r 个模态的POD基

(2a) 一种簇选择算法

$$c\colon \boldsymbol{\rho}(t) \mapsto l \in \{1, 2, \ldots, k\}, \tag{5}$$

(2b) 和 k 线性解码器

$$\tilde{\mathbf{v}} = \mathbf{V}_l(\mathbf{V}_l^\top \mathbf{V})\boldsymbol{\rho},$$

其中，$\mathbf{V}$ 是基于全部数据的POD基，$\mathbf{V}_l \in \mathbb{R}^{n_v \times n_\rho}$ 是与第l个簇对应的POD基，$l = 1$，2，$\cdots$，k且 k 是簇的数量。为构建模型，我们对低维状态向量 $\boldsymbol{\rho}$采用k-均值聚类，并据此对实际数据$\mathbf{v}$ 进行分类，然后分别计算每个簇的POD基。因此，在评估过程中，从降阶状态向量 $\boldsymbol{\rho}(t)$ red中提取标签 $l \in \{1, \ldots, k\}$，并为该标签选择合适的POD基。请注意，聚类是非线性的且不连续的，因此解码也会变成非线性且不连续。

3.5 个体CAE（iCAE）

作为另一种聚类方法，我们将此前为采用基于POD重构的卷积神经网络提出的个体卷积自编码器方法（iCAE）[13] 应用于通用卷积自编码器，如第3.3节所示。一个插值卷积自编码器模型由以下部分组成

(1) 非线性卷积编码器

$$\mu\colon \mathbf{v}_{\mathsf{CNN}} \to \boldsymbol{\rho}$$

如 第3.3节(1) 所示，(2a) 中的聚类操作如 第3.4节(2a) 所示，以及

(2b) k 仿射线性反卷积解码器 rs

$$\tilde{\mathbf{v}} = \phi_l(\boldsymbol{\rho}),$$

其中，ϕ_l 是由第 l个簇训练的第 l个仿射线性反卷积解码器。编码器是连续且可微的。另一方面，解码是非线性且不连续的。众所周知，数据压缩可以提高聚类准确率。由于 CAE 在训练 iCAE 之前已经完成训练，因此无需额外训练即可使用预训练编码器，并且可以将降阶低维状态 $\boldsymbol{\rho}$ 用于 k-均值聚类。因此，在冻结编码器参数的同时，我们仅基于k 个簇训练 k 个解码器。在推理过程中，编码器提取降阶状态向量后，会同时得到一个降阶向量 $\boldsymbol{\rho}$ 和一个标签，然后根据该标签选择适当的解码器。

4 状态与模型的低阶参数化

在本节中，我们应用第 3 节所述模型，以 ODE 形式 (2) 定义不可压缩纳维-斯托克斯方程的低维 LPV 近似。在适用的情况下，即解码为线性的情况下，我们将阐述 LPV 近似的仿射线性结构，该结构

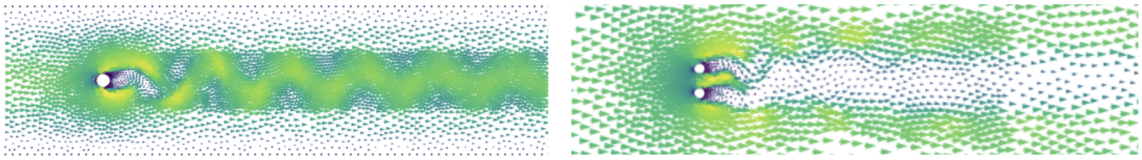

图2：已发展速度状态的两个快照：（左）Re=40 下的单圆柱情形；（右）Re=50下的双圆柱算例

由纳维–斯托克斯方程的二次特性自然导出，因为$\mathbf{v} \mapsto \mathbf{N}(\mathbf{v})$ 是线性的；另请参阅 [12]。我们主要关注非线性项的低维参数化近似。尽管如此，对于采用可微解码器的方法，我们也给出降阶坐标下的方程。具体而言，如前所述，聚类方法会产生不连续的重构 $\tilde{\mathbf{v}}$，因此 $\tilde{\mathbf{v}}$ 可能未定义。

两个测试案例的残差数值计算结果将在第 5 节中报告。

1. 对于**POD** 编码器，参数化形式为

$$\mathbf{M}\dot{\mathbf{v}} + \Pi^\top[\big(\sum_{i=1}^{n_\rho} \rho_i(\mathbf{v})\mathbf{N}(\mathbf{w}_i)\big)\mathbf{v} + \mathbf{A}\mathbf{v} - \mathbf{f}] = 0$$

其中 $\boldsymbol{\rho}(\mathbf{v}) = \mathbf{V}^\top\mathbf{v}$，且$\mathbf{w}_i$ 是 iPOD 基$\mathbf{V}$ 中按第 3.1 节定义的第

$$\hat{\mathbf{M}}\dot{\boldsymbol{\rho}} + \Pi^\top[\big(\sum_{i=1}^{n_\rho} \rho_i\hat{\mathbf{N}}(\mathbf{w}_i)\big)\boldsymbol{\rho} + \hat{\mathbf{A}}\boldsymbol{\rho} - \mathbf{f}] = 0, \tag{6}$$

其中 $\hat{}\mathbf{M} := \mathbf{M}\mathbf{V}$，$\hat{}\mathbf{A} := \mathbf{A}\mathbf{V}$，且 $\hat{}\mathbf{N}(\cdot) := \mathbf{N}(\cdot)\mathbf{V}$。另外请注意，标准的 POD 降阶模型可由式（6）左乘$\mathbf{V}^\top$ 得到。

2. 对于卷积神经网络方法，参数化方程为

$$\mathbf{M}\dot{\mathbf{v}} + \Pi^\top[\big(\sum_{i=1}^{n_\rho} \rho_i(\mathbf{v})\mathbf{N}(\mathbf{w}_i)\big)\mathbf{v} + \mathbf{A}\mathbf{v} - \mathbf{f}] = 0$$

其中 $\boldsymbol{\rho}(\mathbf{v}) = \mu(\mathbf{v})$, 且 $\mathbf{w}_i$ 是 iVW 的第 **列向量**；参见 第 3.2 节。降维坐标下的方程通过以下方式得到

$$\hat{\mathbf{M}}\dot{\boldsymbol{\rho}} + \Pi^\top[\big(\sum_{i=1}^{n_\rho} \rho_i\hat{\mathbf{N}}(\mathbf{w}_i)\big)\boldsymbol{\rho} + \hat{\mathbf{A}}\boldsymbol{\rho} - \mathbf{f}] = 0$$

其中 $\hat{}\mathbf{M} := \mathbf{M}\mathbf{V}\mathbf{W}$，$\hat{}\mathbf{A} := \mathbf{A}\mathbf{V}\mathbf{W}$，且 $\hat{}\mathbf{N}(\cdot) := \mathbf{N}(\cdot)\mathbf{V}\mathbf{W}$。

3. 对于卷积自编码器模型，参数化方程为

$$\mathbf{M}\dot{\mathbf{v}} + \Pi^\top[(\mathbf{N}(\mathbf{b}) + \sum_{i=1}^{n_\rho} \rho_i(\mathbf{v})\mathbf{N}(\mathbf{d}_i))\mathbf{v} + \mathbf{A}\mathbf{v} - \mathbf{f}] = 0$$

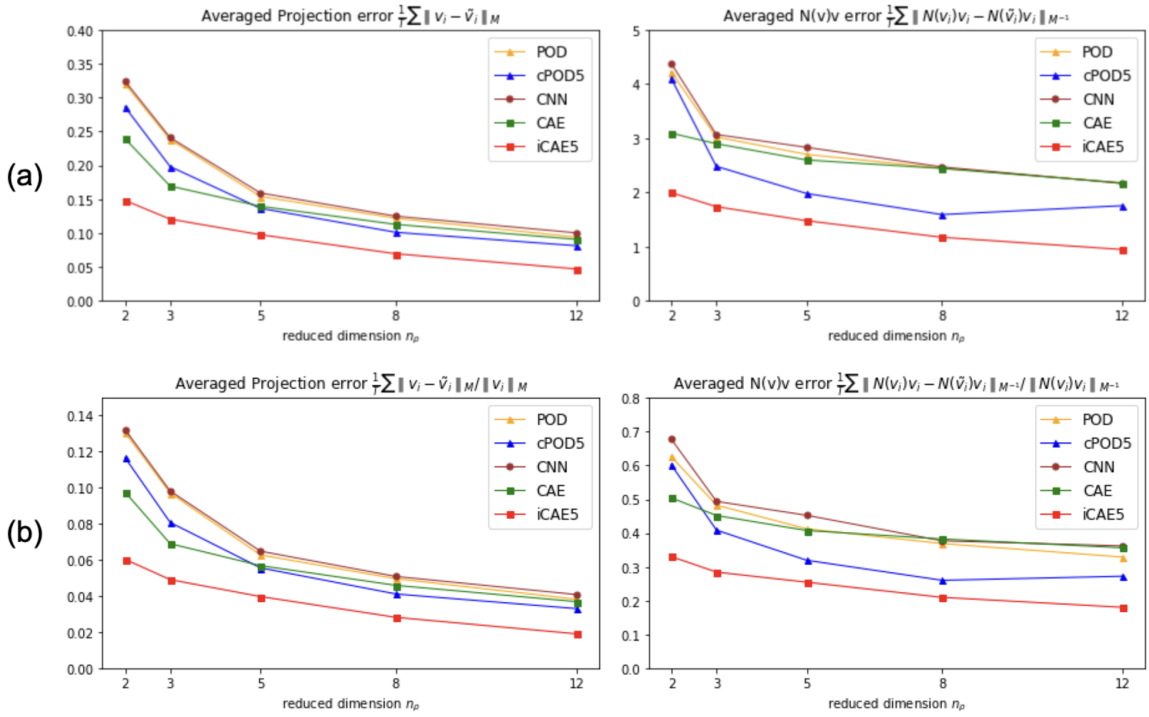

图 3：(a) 单圆柱情形的平均重构误差 (b) 单圆柱情形的平均相对误差（见第 5.1 节）。

其中 $\boldsymbol{\rho}(\mathbf{v}) = \mu(\mathbf{v})$，且$\mathbf{b}$ 是解码器的偏置，$\mathbf{d}_i$ 是 i-th 列向量，属于$\mathbf{D}$；参见第3.3节。降维坐标中的方程通过以下方式得到

$$\hat{\mathbf{M}}\dot{\boldsymbol{\rho}} + \Pi^\top[(\mathbf{N}(\mathbf{b}) + \sum_{i=1}^{n_\rho} \rho_i \mathbf{N}(\mathbf{d}_i))(\mathbf{D}\boldsymbol{\rho} + \mathbf{b}) + \hat{\mathbf{A}}\boldsymbol{\rho} + \hat{\mathbf{f}}] = 0$$

$\hat{}\hat{}\hat{}$ 其中 $\mathbf{M} := \mathbf{M}\mathbf{D}$，$\mathbf{A} := \mathbf{A}\mathbf{D}$，以及 $\mathbf{f} := \mathbf{A}\mathbf{b} - \mathbf{f}$。

4. 对于 **卷积本征正交分解** 模型，参数化方程为

$$\mathbf{M}\dot{\mathbf{v}} + \Pi^\top[(\sum_{i=1}^{n_\rho} \rho_i(\mathbf{v})\mathbf{N}(\mathbf{w}_i^l))\mathbf{v} + \mathbf{A}\mathbf{v} - \mathbf{f}] = 0$$

其中 $\boldsymbol{\rho}(\mathbf{v}) = \mathbf{V}^\top\mathbf{v}$ 和 $\mathbf{w}_i^l$ 是 i矩阵 $\mathbf{V}_l(\mathbf{V}_l^\top\mathbf{V})$ 的第 $\mathbf{V}$ 整体 POD 基中的 POD 基 $\mathbf{V}_l$，属于相应簇；参见第3.4节。

5. 对于 **插值卷积自编码器** 模型，参数化方程为

$$\mathbf{M}\dot{\mathbf{v}} + \Pi^\top[(\mathbf{N}(\mathbf{b}_l) + \sum_{i=1}^{n_\rho} \rho_i(\mathbf{v})\mathbf{N}(\mathbf{d}_i^l))\mathbf{v} + A\mathbf{v} - \mathbf{f}] = 0$$

其中 $\boldsymbol{\rho}(\mathbf{v}) = \mu(\mathbf{v})$，$\mathbf{b}_l$ 是第 l 个解码器的偏置，而 $\mathbf{d}_i^l$ 是 i 的第 $\mathbf{D}_l$个列向量；参见第 3.5节。

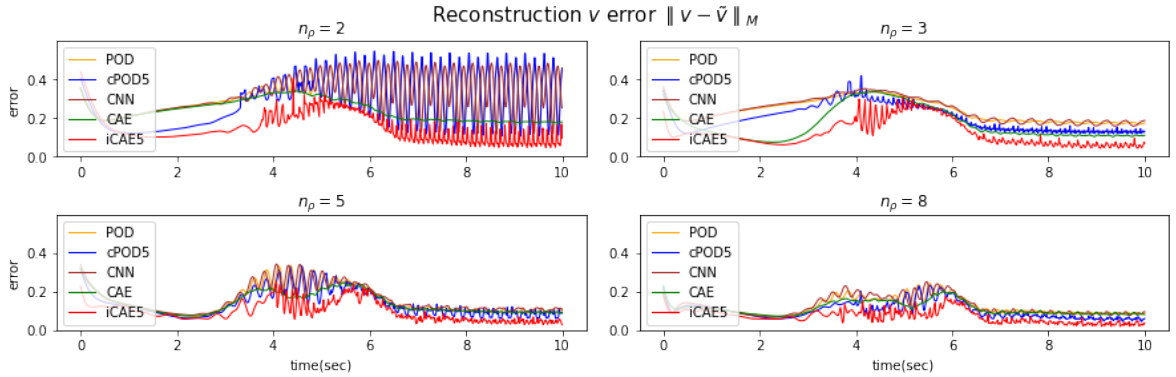

图 4：单圆柱情形（ $[0, 10]$ 下的重构误差（第 5.1 节）。

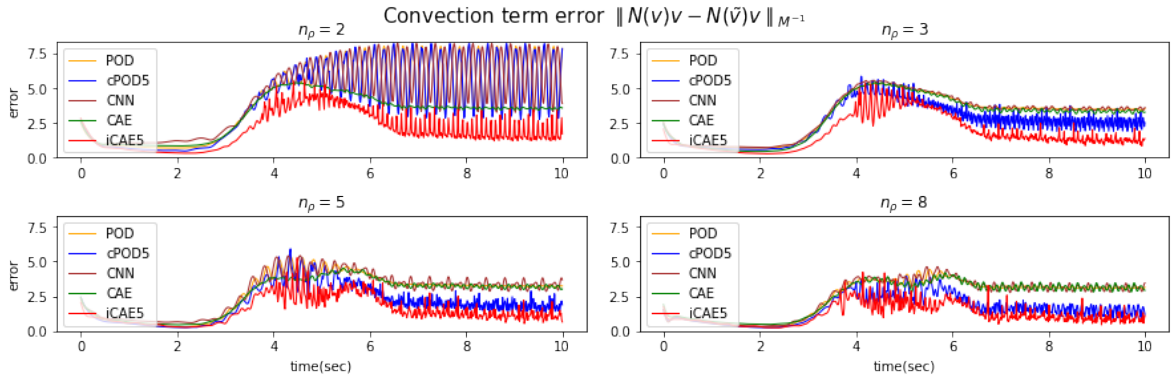

图 5：单圆柱情形（ $[0, 10]$ 下的对流误差（第 5.1 节）。

至于 **iCAE** 和 **cPOD** 模型，我们注意到，通过标记算法（ $l = l(\boldsymbol{\rho})$ 对 $\boldsymbol{\rho}$ 的依赖（5）使解码过程产生了非线性。尽管如此，由于 $n_\rho \cdot k$ 系数 $\mathbf{N}(\mathbf{w}_i^l)$ 或 $\mathbf{N}(\mathbf{d}_i^l)$ 可以预先计算， $i = 1, \ldots, n_\rho$, $l = 1, \ldots, k$，其中 k 是簇的数量，因此可以利用降阶模型固有的仿射线性结构，在数值模拟中实现高效计算。

5 数值示例

圆柱尾流是一个成熟的基准示例，其流动模式会随模拟设置和雷诺数而变化。本节分别研究如图2所示的单圆柱和双圆柱后方出现的二维圆柱尾流现象。我们利用包含单圆柱后方周期性尾流的数据，以及包含双圆柱后方混沌涡旋的数据，考察模型如何生成重构状态向量。此外，本研究的主要主题是探究不可压缩流的极低维参数化，因此采用了 $n_\rho = 2$、3、5、8、12。

原始数据、所有程序以及用于复现所展示结果的脚本，均已公开发布；详情见论文末尾的第7节。

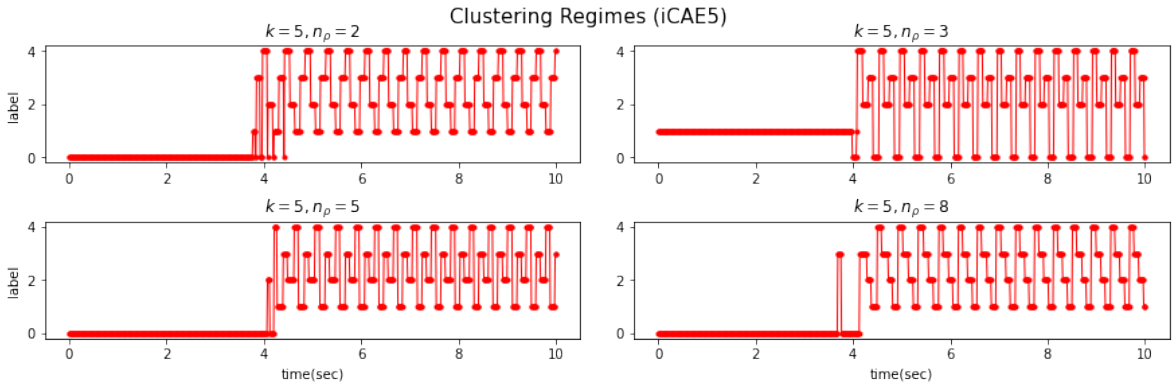

图 6：单圆柱情形下的提示聚类反应 $[0, 10]$ （第 5.1 节）。

为了评估所有实验，我们使用以下六项评估指标：

- 平均重构误差

$$\frac{1}{T}\sum_{i=1}^{T} \| \mathbf{v}_i - \tilde{\mathbf{v}}_i \|_{\mathbf{M}},$$

- 平均重构相对误差 $\mathbf{v}$

$$\frac{1}{T}\sum_{i=1}^{T} \| \mathbf{v}_i - \tilde{\mathbf{v}}_i \|_{\mathbf{M}} \; / \; \| \mathbf{v}_i \|_{\mathbf{M}}$$

- 平均对流项误差,

$$\frac{1}{T}\sum_{i=1}^{T} \| \mathbf{N}(\mathbf{v}_i)\mathbf{v}_i - \mathbf{N}(\tilde{\mathbf{v}}_i)\mathbf{v}_i \|_{\mathbf{M}^{-1}},$$

- 平均重构相对误差 $\mathbf{N(v)v}$

$$\frac{1}{T}\sum_{i=1}^{T} \| \mathbf{N}(\mathbf{v}_i)\mathbf{v}_i - \mathbf{N}(\tilde{\mathbf{v}}_i)\mathbf{v}_i \|_{\mathbf{M}^{-1}} \; / \; \| \mathbf{N}(\mathbf{v}_i)\mathbf{v}_i \|_{\mathbf{M}^{-1}},$$

- 平均残差

$$\frac{1}{T}\sum_{i=1}^{T} \| \Pi^{\top}[(\mathbf{N}(\tilde{\mathbf{v}})\tilde{\mathbf{v}} + \mathbf{A}\tilde{\mathbf{v}}) - (\mathbf{N}(\mathbf{v})\mathbf{v} + \mathbf{A}\mathbf{v})] \|_{\mathbf{M}^{-1}},$$

- 和平均相对残差

$$\frac{1}{T}\sum_{i=1}^{T} \| \Pi^{\top}[(\mathbf{N}(\tilde{\mathbf{v}})\tilde{\mathbf{v}} + \mathbf{A}\tilde{\mathbf{v}}) - (\mathbf{N}(\mathbf{v})\mathbf{v} + \mathbf{A}\mathbf{v})] \|_{\mathbf{M}^{-1}} \; / \; \| \Pi^{\top}(\mathbf{N}(\mathbf{v})\mathbf{v} + \mathbf{A}\mathbf{v}) \|_{\mathbf{M}^{-1}},$$

其中 $\| \mathbf{x} \|_{\mathbf{M}} = \sqrt{\mathbf{x}^{\top}\mathbf{M}\mathbf{x}}$，$\mathbf{v}_i$ 是目标速度，$\tilde{\mathbf{v}}_i$ 是重构速度，T 是快照数量。

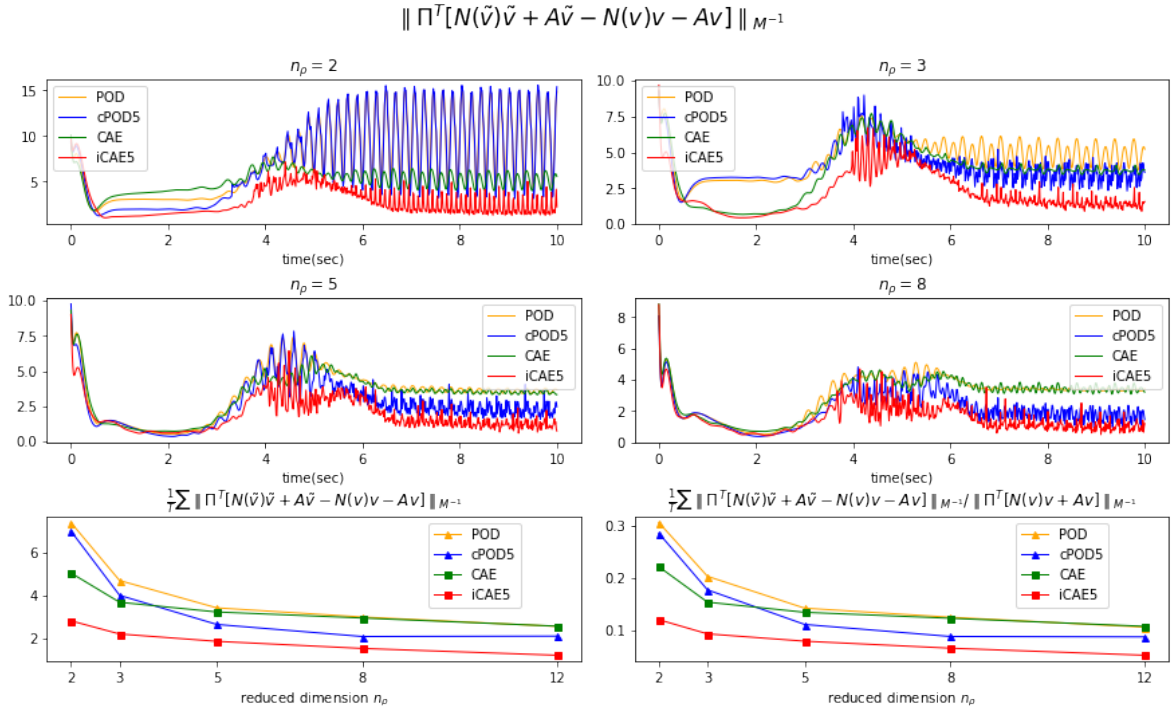

图 7：误差 $\Pi^{\top}[\mathbf{N}(\tilde{\mathbf{v}})\tilde{\mathbf{v}} + \mathbf{A}\tilde{\mathbf{v}} - (\mathbf{N}(\mathbf{v})\mathbf{v} + \mathbf{A}\mathbf{v})]$ 见于 $[0, 10]$ (第 5.1 节)。

5.1 单圆柱

在训练方面，也就是建立POD基并训练神经网络和聚类算法时，我们使用了 x 和 y 方向速度在 $[0, 10]$中的 400 个等距快照。这些快照由有限元模拟生成，模拟对象为雷诺数为 40 的圆柱尾流，模拟方程为（1），并以某个不太符合物理实际的相应斯托克斯解作为初始条件。所有数据点 $\mathbf{v}(t) \in \mathbb{R}^{42764}$(即 $n_v = 42764$)在区域 (0,5) $\times$ (0,1)内进行空间分布。为将数据输入卷积神经网络，使用插值矩阵 $\mathbf{v}_{\mathsf{CNN}}(t) \in \mathbb{R}^{2\times 47\times 63}$ 生成卷积神经网络输入数据 $\mathbf{I}_c \in \mathbb{R}^{42764\times 5922}$，随后将数据向量重塑为三维张量。对于评估数据集，采用相同条件在 $[0, 10]$上生成了 800 个快照。

CNN/CAE 模型的参数列于表 1。基本上，卷积神经网络模型的编码器包含四个卷积层和一个全连接层，解码器则使用线性变换和一个 POD基 $\mathbf{V} \in \mathbb{R}^{42764\times 15}$ (即 $r = 15$)，将降阶参数映射回原始状态空间。

卷积自编码器的编码器包含三个卷积层和一个全连接层。解码器依次执行一个全连接层、三个反卷积层和一个全连接层。在编码器部分，每个隐藏层都应用了非线性激活函数ELU [6] 。对于聚类模型，簇的数量 k 为 5，所考虑的聚类模型分别称为 cPOD5 和 iCAE5。

在网络的训练过程中，我们使用 ADAM 优化器 [15] 和多步学习率调度器，批量大小为 64。损失函数采用均方误差。使用$\mathbf{v}_{\mathsf{CNN}}$ 数据训练卷积自编码器时，我们使用满足$\mathbf{v} = \mathbf{I}_p\mathbf{v}_{\mathsf{CNN}}$ 的插值矩阵$\mathbf{I}_p$，将状态从张量化网格映射回有限元网格。需要注意的是，由于张量化网格相当粗糙，两次插值会导致大量细节丢失，尤其是$\mathbf{v} \neq \mathbf{I}_p\mathbf{I}_c\mathbf{v}$。

由于使用$\mathbf{I}_p$ 和$\mathbf{I}_c$ 会造成显著的精度损失，因此可以尝试改为学习这些

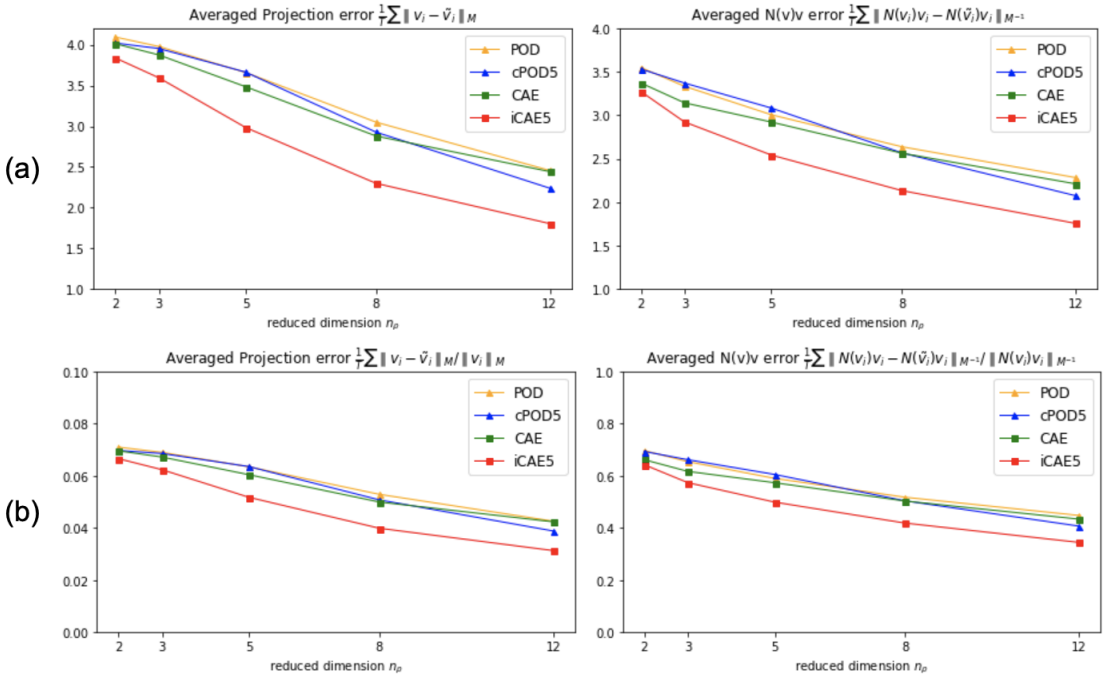

图 8：双圆柱算例的（a）平均重构误差和（b）平均相对误差（第5.2节）。

插值操作。因此，我们同时优化解码器最后一层中的$\mathbf{I}^p$ 及其他解码器参数，而不使用通过优化$\mathbf{v} \approx \mathbf{I}^p \mathbf{v}_{\text{CNN}}$得到的固定$\mathbf{I}^p$ 。

图 3 展示了卷积本征正交分解和插值卷积自编码器如何分别提升标准 POD 和卷积自编码器的重构性能。卷积神经网络模型与 POD 的结果相近。在极低维的情形下（$n_\rho = 2,\ 3$），卷积自编码器的表现优于卷积本征正交分解。关于对流项误差，除了卷积本征正交分解外，观察到的趋势与投影误差相似。卷积本征正交分解的误差曲线在 $n_\rho = 12$处上升，与随着 n_ρ 增大而下降的趋势相反。 图 4显示，当 $n_\rho = 2$时，POD 和卷积本征正交分解的性能会大幅下降。总体而言，插值卷积自编码器随时间推移表现最佳。然而，当 $n_\rho = 5$时，在约 4 秒处会出现峰值。

图 5 表明，$\mathbf{N}(\mathbf{v})\mathbf{v}$ 误差往往与重构$\mathbf{v}$ 误差相似，而且预训练模型能够同样出色地逼近$\mathbf{N}(\mathbf{v})\mathbf{v}$ ，尽管实际的对流损失$L = \sum_{i=1}^{T} \| \mathbf{N}(\mathbf{v}_i)\mathbf{v}_i - \mathbf{N}(\tilde{\mathbf{v}}_i)\mathbf{v}_i \|_{\mathbf{M}^{-1}}$ 并未用于训练过程。

一般而言，我们观察到，考虑有限元范数（例如使用 $L = \sum_{i=1}^{T} \| \mathbf{v}_i - \tilde{\mathbf{v}}_i \|_2^2$ 计算损失）并未改善重构误差，反而带来了显著的计算开销。因此，我们仍采用标准的均方误差损失函数，它对应于表示状态$\mathbf{v}$ 以及形如$\mathbf{N}(\mathbf{v})$的向量的非加权二范数。

对于 iCAE 方法，一旦编码器μ 可用，便会学习一种针对速度状态的分类策略。图 6 显示，iCAE5 能够捕捉周期性流动模式，即圆柱体后方四秒处的尾流。从字面上说，数据是构建数据驱动模型最重要的因素。换言之，聚类会影响重构性能，因为解码器是使用聚类数据集训练的。当状态被错误分类时，可能会导致瞬间出现高误差尖峰。根据图 4，与 iCAE 的整体性能相反，观察到若干异常

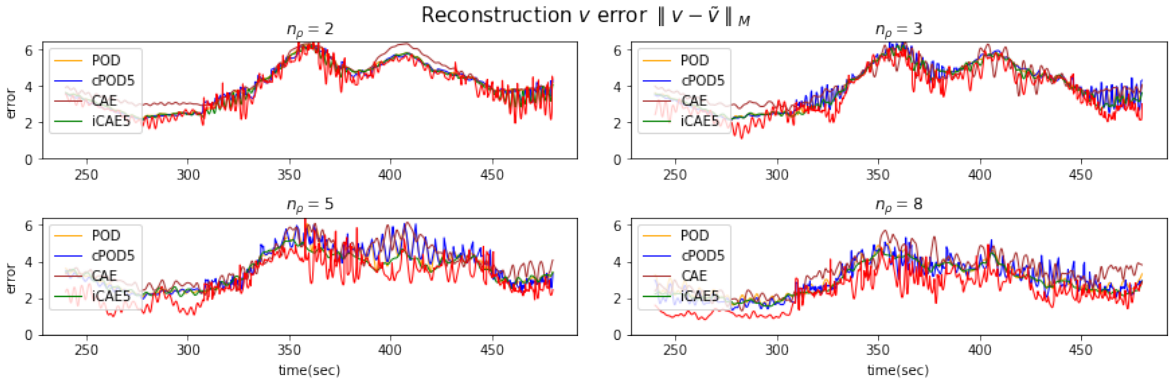

图9：重构误差，见 $[240, 480]$ （第5.2节）。

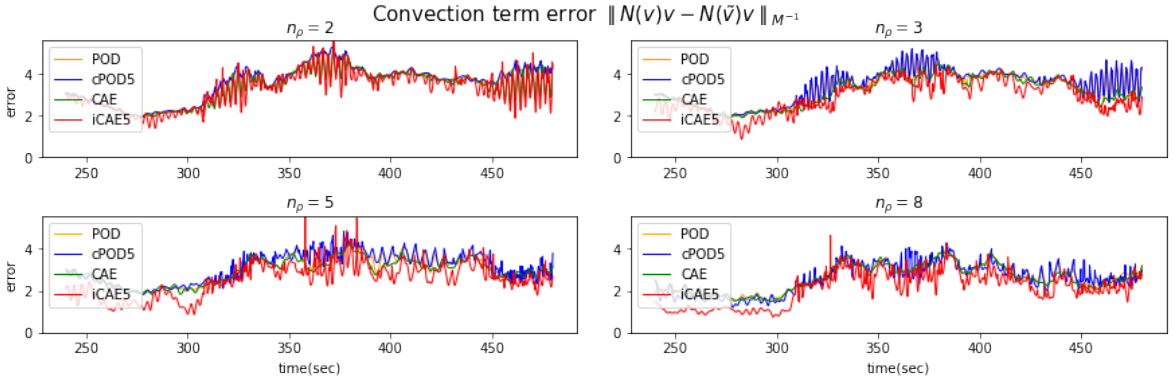

图10：双圆柱算例中的平均对流误差，见 $[240, 480]$ （第5.2节）。

在时间段 $[4, 5]$ 当 $n_\rho = 5$时，观察到了这些值。

接下来，我们使用残差指标比较残差，而不借助时间导数。如第4节所述，聚类模型无法重现 $\tilde{\mathbf{v}}$。此外，由于数据的计算采用了复杂的二阶时间积分格式，因此，仅基于快照对 $\dot{v}$ 进行数值计算，而这些快照对时间域的覆盖仅是粗略的，其误差完全主导了残差。在图 7中，误差呈现出与图 3所示重构误差相似的趋势：插值卷积自编码器表现最佳，而在较小的 ρ维度下，卷积自编码器的线性重构优于 POD。另请注意图 4 和图 5中的残差轨迹图，它们清楚地显示了这些趋势；但在初始阶段，所有方法的近似结果都相近。

5.2 双圆柱

作为一个动力学更丰富的第二个示例，我们考虑雷诺数为 50 的双圆柱尾流方程（1）的有限元模拟；设置示意图见 图2 。这里，状态向量 $\mathbf{v}(t) \in \mathbb{R}^{25982}$ （即 $n_v = 25982$）分布在物理域 $(-20,\ 70) \times (-20,\ 20)$。因此，为了获得卷积神经网络输入数据 $\mathbf{v}_{\mathsf{CNN}}(t) \in \mathbb{R}^{2 \times 47 \times 63}$，需要应用插值矩阵 $\mathbf{I}_c \in \mathbb{R}^{25982 \times (47 \cdot 63)}$。

用于训练　这些模型中，我们使用了时间区间内 800 个快照，包含 x 和 y 方向速度　1

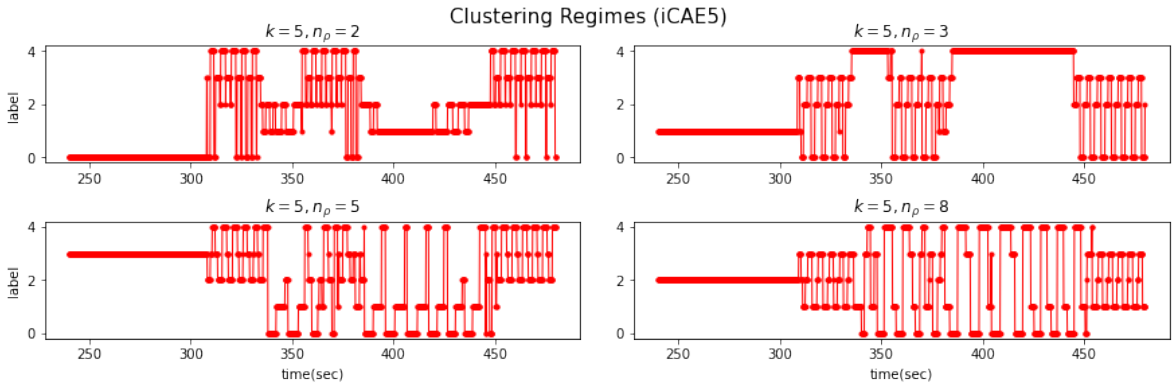

图11：双圆柱算例中的提示聚类反应 $[240, 480]$ （第5.2节）。

$[240, 480]$, 因此忽略了启动阶段。

在评估阶段，使用了1600个快照，其条件与$[240, 480]$ 中的训练数据集相同。其余实验设置与单圆柱情形相同。由于卷积神经网络模型（3）表现不具竞争力，因此将其排除在实验之外，随后将具有相同超参数的现有模型应用于双圆柱算例。

平均误差（参见图 8）证实了前一个示例的结论，即插值卷积自编码器取得了最佳结果，并且卷积自编码器的表现优于POD，尤其是在参数维度较低时。然而，与单圆柱情形不同，图9和图10中展示的随时间变化的误差并未呈现明显差距。这可能尤其是由于流动存在许多不同的状态；这似乎也是与 ρ的不同维度进行比较时，聚类结果未显示明显规律的原因；参见图11，其中iCAE5根据 $n_\rho = 2$、3、5、8得出不同的聚类结果。尽管如此，如图12所示，插值卷积自编码器在 $\Pi^\top(\mathbf{N}(\mathbf{v})\mathbf{v}+\mathbf{A}\mathbf{v})$ 的降维维度n_ρ上，残差也达到最低。

6 结论

本文研究了 POD、卷积自编码器和聚类的多种组合，用于流体流动的低维参数化。我们证实，正如大量理论研究和数值研究所指出的，在参数化维度极低时，非线性方法的表现可以优于 POD 等线性方法。

此外，我们证实，神经网络提供了一种通用方法，即使在高维数据情形下也能找到这些低维参数化；尤其是当能够利用卷积神经网络中隐含包含的稀疏模式时。

为了探究参数化的极限，我们在降维坐标上采用了 k-均值聚类，使重构能够针对特定流动状态进行。尽管这种方法（就本文中的插值卷积自编码器方法而言）能够达到显著更低的误差，尤其是在低维情况下，但其实际应用（例如在仿真中）仍存疑，因为解码操作并不光滑。

未来，我们将研究所提出的低维参数化在仿真和控制器设计中的性能。另一项值得关注的未来工作是对聚类映射进行松弛，使基于聚类信息的重构成为平滑、甚至可能是线性的操作。

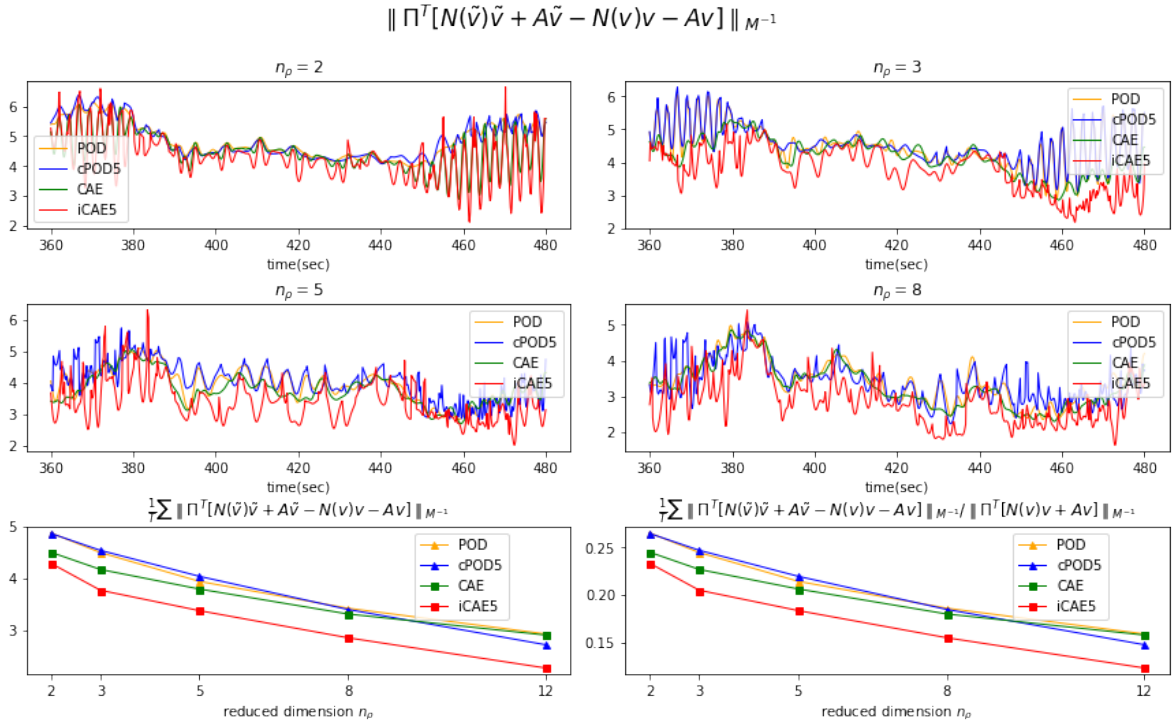

图12： $\Pi^{\top}(\mathbf{N}(\mathbf{v})\mathbf{v} + \mathbf{A}\mathbf{v} - \mathbf{f})$ 在 $[360, 480]$ (第5.2节）。

7 代码可用性

用于计算本文所呈现结果的实现源代码可从doi:10.5281/zenodo.7575808 获取。该代码采用知识共享署名 4.0 国际许可协议，并由 Yongho Kim 编写。

致谢

我们感谢德国研究联合会（DFG）通过马格德堡的研究生培训小组2297“MathCoRe”提供资助。

参考文献

[1] R. Altmann 和 J. Heiland。流动方程的有限元分解与最小延拓。《ESAIM 数学建模与数值分析》，49(5):1489–1509，2015年9月。doi:10.1051/ m2an/2015029。[2] M. Balcan 和 K. Q. Weinberger，编者。《用于聚类分析的无监督深度嵌入》，JMLR研讨会与会议论文集第48卷。JMLR.org，2016年。URL: http:// proceedings.mlr.press/v48/xieb16.html。[3] D. Bank、N. Koenigstein 和 R. Giryes。自编码器。arXiv，2020年。URL: https://arxiv.org/ abs/2003.05991。[4] M. Behr、P. Benner 和 J. Heiland。带控制与观测的纳维–斯托克斯方程示例设置：空间离散化及通过线性二次矩阵系数进行表征。ArXiv，2017年。URL: https://arxiv.org/abs/1707.08711。[5] G. Berkooz、P. Holmes 和 J. L. Lumley。湍流分析中的本征正交分解。《流体力学年度评论》，25(1):539–575，1993年。doi:10.1146/annurev.fl.25. 010193.002543。[6] D.-A. Clevert、T. Unterthiner 和 S. Hochreiter。利用指数线性单元（ELU）实现快速、准确的深度网络学习。国际学习表征会议 2016（海报），2016年。URL: https://arxiv.org/pdf/ 1511.07289.pdf。[7] M. M. Fard、T. Thonet 和 ´E. Gaussier。深度 k-均值：结合 k-均值聚类与表征学习。《模式识别快报》，138:185–192，2020年。doi:10.1016/j.patrec. 2020.07.028。[8] S. Fresca 和 A. Manzoni。正交本征分解深度学习降阶模型：通过本征正交分解增强用于非线性参数化偏微分方程的深度学习降阶模型。《计算方法与应用力学及工程》，388:114181，2022年。doi:10.1016/j.cma.2021.114181。[9] H. Gao、L. Sun 和 J. Wang。PhyGeoNet：用于求解非规则区域上参数化稳态偏微分方程的物理信息驱动几何自适应卷积神经网络。《计算物理学杂志》，428:110079，2021年。doi:10.1016/j.jcp.2020.110079。[10] J. Gao 和 J. Zhang。潜在语义索引中的聚类奇异值分解策略。《信息处理与管理》，41(5)，2005年。doi:10.1016/j.ipm.2004.10.005。[11] I. J. Goodfellow、Y. Bengio 和 A. Courville。《深度学习》。麻省理工学院出版社，美国马萨诸塞州剑桥，2016年。URL: http://www.deeplearningbook.org。[12] J. Heiland、P. Benner 和 R. Bahmani。用于不可压缩纳维–斯托克斯方程超低维线性参数变化近似的卷积神经网络。《应用数学与统计学前沿》，8:879140，2022年。doi:10.3389/fams.2022.879140。[13]J. Heiland 和 Y. Kim。用于不可压缩流动低维参数化的卷积自编码器与聚类。IFAC-PapersOnLine，55(30):430–435，2022年。第25届MTNS国际研讨会论文集。doi:10.1016/j.ifacol.2022.11.091。[14] Y. Kim 和 Y. Choi。使用全连接神经网络学习反应–扩散型方程的有限差分方法。《计算数学与……》，123:115–122，2022年。doi:10.1016/j.camwa.2022.08.006。[15] D. P. Kingma 和 J. Ba。Adam：一种随机优化方法。国际学习表征会议 2015，会议论文集，2015年。URL: http://arxiv.org/abs/1412.6980。

[16] P. J. W. Koelewijn 和 R. T´oth. 线性参数时变模型的维数约简调度——一种深度神经网络方法。IEEE会议录，第 1111–1117 页，2020年。doi:10.23919/ACC45564. 2020.9147310.[17] Y. LeCun、L. Bottou、Y. Bengio 和 P. Haffner. 应用于文档识别的基于梯度的学习。《IEEE会议录》，86(11):2278–2324，1998年。doi:10.1109/5.726791.[18] K. Lee 和 K. T. Carlberg. 利用深度卷积自编码器对非线性流形上的动力系统进行模型降阶。《计算物理学杂志》，404，2020年。doi:10.1016/j.jcp.2019. 108973.[19] D. Liu、S. Xie、Y. Li、D. Zhao 和 E. M. El-Alfy（编）。利用卷积自编码器进行深度聚类，计算机科学讲义第 10635 卷。Springer，2017年。doi:10.1007/978-3-319-70096-0_39.[20] M. Ohlberger 和 S. Rave. 降阶基方法：成效、局限性与未来挑战。《Algoritmy 会议论文集》，第 1–12 页，2016年。URL: https://arxiv.org/abs/ 1511.02021.[21] S. Z. Rizvi、F. Abbasi 和 J. M. Velni. 利用自编码器神经网络对线性参数时变模型进行模型降阶。IEEE，第 6415–6420 页，2018年。doi:10.23919/ACC.2018.8431912.[22] A. Saxena、M. Prasad、A. Gupta、N. Bharill、O. P. Patel、A. Tiwari、M. J. Er、W. Ding 和 C. Lin. 聚类技术及其发展的综述。《神经计算》，267:664–681，2017年。doi:10.1016/j.neucom.2017.06.053.[23] D. Sculley. Web规模k均值聚类。WWW'10，第 1177–1178 页，2010年。doi:10.1145/1772690.1772862.[24] N. Winovich、K. Ramani 和 G. Lin. ConvPDE-UQ：用于变化域上异质椭圆偏微分方程的、具有量化不确定性的卷积神经网络。《计算物理学杂志》，394:263–279，2019年。doi:10.1016/j.jcp.2019.05.026。