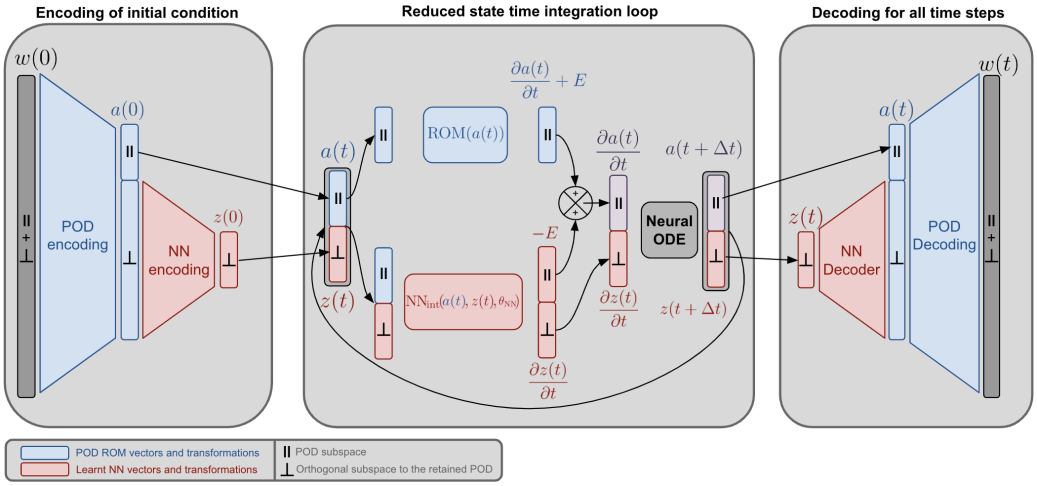
Graphical Abstract

Hybrid AutoEncoder/Galerkin approach for nonlinear reduced order modelling

Nicolas Lepage, Samir Beneddine, Camilla Fiorini, Iraj Mortazavi, Denis Sipp, Nicolas Thome



Highlights

Hybrid AutoEncoder/Galerkin approach for nonlinear reduced order modelling

Nicolas Lepage, Samir Beneddine, Camilla Fiorini, Iraj Mortazavi, Denis Sipp, Nicolas Thome

- Hybrid physics/deep learning reduced model for accurate and interpretable prediction
- Modelling and reconstruction of smallest useful scales that are usually discarded
- Neural ODE time integration aligns the model with underlying physics
- Experiments conducted on nonlinear one-dimensional and two-dimensional test cases
- Experiments validate model performance and robustness compared to existing approaches

Hybrid AutoEncoder/Galerkin approach for nonlinear reduced order modelling

Nicolas Lepage[a,*], Samir Beneddine[b], Camilla Fiorini[a], Iraj Mortazavi[a], Denis Sipp[b], Nicolas Thome[c]

[a] *CNAM, M2N lab, Paris, 75003, France*
[b] *ONERA, DAAA, Institut Polytechnique de Paris, Meudon, 92190, France*
[c] *Sorbonne University, ISIR, Paris, 75005, France*

Abstract

This paper presents a novel nonlinear Reduced Order Model (ROM) that combines Proper Orthogonal Decomposition (POD) with deep learning residual error correction. Deep learning is used for error correction in both the projection and time integration phases of the ROM. This enables simultaneous correction within the POD subspace (error in the reduced subspace) and outside (truncation error). The present hybrid ROM is trained using an end-to-end neural Ordinary Differential Equations (ODE) framework, aligning the deep learning component with the continuous-time nature of the governing equations. We evaluate its performance using well-studied test cases: the viscous Burgers equation, the cylinder flow at a single Reynolds number (equal to 100), as well as for Reynolds numbers ranging from 60 to 120 (parametric cylinder case) and the fluidic pinball in the quasi-periodic regime. These non-chaotic test cases, are chosen to assess different aspects of the method and its ability to accurately predict reproducible dynamics. Our novel strategy outperforms several existing approaches both in terms of accuracy and dimensionality reduction: POD Galerkin ROMs, a purely data-driven approach using only autoencoders, and also state-of-the-art hybrid methods. Furthermore, it offers low computational overhead compared to classical POD-based ROMs, making it attractive for complex 2D or 3D systems.

Keywords: Autoencoder, Reduced order model, Neural ODE,

*Corresponding author.

1. Introduction

Fluid dynamics is crucial in various industries, including automotive, aviation, and weather forecasting. Except for trivial cases, closed-form solutions do not exist [1], requiring spatio-temporal discretisations of the Navier-Stokes equations to predict complex flow field behaviour. Although these methods provide accurate predictions, they often demand computational costs exceeding available resources for intricate problems [2], leading to ad-hoc models. This compromises accuracy [3] and relies on parameters that are difficult to determine [4].

Reduced Order Models (ROMs) provide an appealing alternative by reducing governing equations to a lower-dimensional manifold. Among ROM approaches, POD-Galerkin methods have gained considerable attention for their ability to extract energetically dominant modes and yield low-dimensional models that preserve constraints such as boundary conditions, conservation laws, and symmetries. Rooted in the early works of [5] and [6], this methodology decomposes flow fields into a finite set of dominant orthonormal modes via Proper Orthogonal Decomposition (POD), then projects the governing equations onto these modes using Galerkin formalism. As the POD basis optimally captures fluctuation variance around a base state, this approach offers a systematic route to derive ROMs directly from Full Order Models (FOM). However, only the most energetic modes are retained, while lower-energy modes are discarded. Early studies [7] showed that accounting for the influence of neglected modes is essential to ensure dynamical stability. In particular, the introduction of a shift-mode significantly improved the robustness and stability of reduced models [8]. This insight spurred further development of nonlinear Galerkin model calibrations [9, 10], with increasing application to flow control [11, 12]. In parallel, Dynamic Mode Decomposition (DMD) has emerged as a powerful technique for extracting coherent spatiotemporal structures from fluid flows. Introduced by [13], DMD bridges linear stability analysis and dynamic flow behaviour, complementing POD-based methods and broadening the model reduction toolbox. DMD can also be combined with POD (in place of the Galerkin projection step) to yield fully data-driven models [14].

Despite widespread adoption and interpretability, these classical linear approaches face challenges when applied to nonlinear and convective prob-

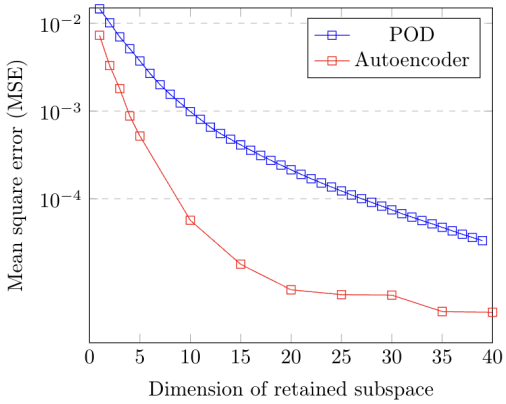


Figure 1: Log-scale plot of the mean squared reconstruction error (MSE) for POD and autoencoder models, computed on the Burgers equation dataset described in Section 5.1. The autoencoder architecture matches that of our hybrid model for the Burgers equation test case (architecture described in Appendix A).

lems in fluid mechanics [15]. In many cases, the linear subspace requires a prohibitively large number of modes to accurately capture nonlinear effects and avoid non-physical wavy modes. Such limitations have driven research aimed at enhancing POD-Galerkin techniques, either through projection procedure modifications or by incorporating additional strategies to better capture nonlinear dynamics.

Recent research has shown that integrating deep learning autoencoders (AE) into linear ROM frameworks may be a promising alternative [15, 16, 17, 18]. Due to their nonlinear nature, AE-based manifolds are an attractive choice for handling nonlinear and convective problems. An introductory numerical experiment compares projection error of POD versus AE depending on retained modes for a dataset generated from viscous Burgers equation simulations (cf. Figure 1). It illustrates AE ability to reach a given error level for far smaller latent spaces than its POD counterpart, demonstrating the method's attractiveness. However, these recent deep-learning approaches have limitations. By construction, they lack interpretability and are not guaranteed to comply with physical laws (conservation laws, boundary conditions, etc) [19, 20]. The nonlinear latent space does not allow easy manifold visualisation or simple projection of discretised operators in the reduced space, meaning that the new model must be learnt in the reduced space or modelling must occur by projecting back to the full subspace.

This paper presents a novel hybrid approach that advances ROMs by integrating the strengths of Proper Orthogonal Decomposition (POD) with auto-encoder (AE) enhancements. Our model offers the following key contributions:

- **Balanced Interpretability and Compression:** We retain the most energetic modes using classical POD, ensuring physical interpretability, whilst the AE encodes less energetic modes, capturing otherwise discarded information in a compact latent space. This balance of interpretability and nonlinear compression improves traditional POD methods. Unlike fully AE-based models, our approach uniquely preserves the physical representativity of the reduced subspace.

- **Time Integration and Hybridisation:** Most dynamics are predicted physically by the ROM, preserving ROM advantages in capturing physical quantities and dynamics. Our learned hybridisation, driven by AE information, is distinct in its Markovian nature and avoids challenges of methods relying on complex, time-dependent corrections based on the Mori-Zwanzig formalism [21] (e.g., LSTMs [22], and temporal convolutions [23, 24]). This simplifies implementation whilst both correcting errors in POD subspace dynamics and modelling the component in the orthogonal subspace.

- **Reconstruction of Discarded Information:** A separate contribution is the hybrid POD-AE latent space, which not only addresses the closure problem for the POD subspace but also reconstructs the dynamics of information typically discarded by POD mode selection. Our method recovers coefficients and dynamics of discarded modes, enhancing accuracy of both dynamics prediction and reconstruction. This allows our model to better capture small-scale phenomena, which would otherwise be lost, significantly improving performance as shown in the results section.

- **Neural ODE for Time-Continuous Integration:** Neural ODEs provide a natural and well-suited framework for our approach, as they inherently align deep learning with the differential nature of physical systems. However, this work focuses on exploring the hybrid nature of our proposed method, rather than evaluating whether Neural ODEs

are the optimal framework. Consequently, we do not specifically assess their impact on accuracy and robustness.

Whilst the results demonstrate that this hybrid design addresses many limitations of purely POD-Galerkin or purely data-driven systems, two practical considerations remain. First, the method remains intrusive: it requires access to discretised operators and is therefore inapplicable when these are unavailable in a given FEM code. Second, although the architecture maintains a clear separation between POD and AE contributions during both dimensionality reduction and dynamics prediction (Section 4), the neural correction can, in principle, introduce arbitrarily large updates that override the physics-based ROM. However, since this correction term is explicit and modular, safeguards such as amplitude clipping or energy-budget constraints can be incorporated to enforce physical admissibility but were not necessary in the four conducted experiments.

2. Related Work

To contextualise this work, this section reviews key literature methods that use deep learning to enhance traditional POD-Galerkin ROMs. We consider two main approaches. The first retains linear POD embedding for state reduction and integrates neural network-based closure models to correct errors within the POD subspace, capturing the effects of truncated modes without explicit representation. The second employs nonlinear encoding through autoencoders, replacing POD with complex, data-driven embeddings, and explores techniques for modelling latent space dynamics, including neural ODEs and recurrent neural networks. We examine these approaches and their variants in detail and finally compare our hybrid model to existing state-of-the-art methods, highlighting key differences in architecture, performance, and computational efficiency.

2.1. Methods using POD based reduction

Methods using only POD-based projection adhere closely to traditional POD-Galerkin ROM methodology. Due to linear encoding, intrusive POD-Galerkin ROMs can still be used, as they access the operator matrix from the numerical solver and project the operator into the reduced subspace. As previously mentioned, these intrusive approaches have several advantages. However, the reduction achieved is generally suboptimal. Despite having

a model in the reduced subspace, the resulting ROM will not account for coupling terms between unretained and retained information. All proposed approaches are formulated as closure models to account for discarded physics effects on reduced system dynamics. Deep learning can model complex nonlinear phenomena and therefore seems suitable for this low-dimensional closure. This general framework is common for all methods using POD solely for dimensionality reduction. However, closure modelling approaches vary; we now present the three main approaches.

Closure modelling via reduced-space dynamics. A distinct class of closure modelling methods seeks to account for the influence of discarded modes on resolved dynamics without reconstructing their state. These approaches instead apply physically inspired or data-driven corrections within the retained POD subspace. A representative overview is given in [25], which compares four physically motivated closure models—originally developed for large eddy simulation—adapted to operate within the retained POD space. More recent extensions, such as [26], replace physics-based closures with neural networks. Both works adopt a two-tier Variational Multiscale (VMS) formulation, where the retained POD basis is split into large and small scales, and the closure term acts exclusively on the small-scale component. Due to the structural similarity with our two-tier POD strategy, we focus here on VMS-type approaches. Our method also partitions the POD space into two blocks, but with a different aim. In VMS, the decomposition is used to confines closure modelling to the lowest-energy retained modes. In contrast, our decomposition targets increased information density via further compression in the latent space. Moreover, unlike VMS, our correction terms can act on both high- and low-energy POD modes. This flexibility proves useful when dominant POD modes alone poorly approximate the system's true dynamics, as in strongly parametric regimes or under severe under-resolution.

Mori-Zwanzig based closure modelling. Many methods leverage the Takens theorem and Mori-Zwanzig formalism to address closure modelling in ROMs. The Takens theorem states that system dynamics can be reconstructed from time-delayed observations of a single variable, whilst the Mori-Zwanzig formalism provides a framework for describing the evolution of variable subsets in dynamical systems, accounting for unresolved variables through memory terms. Applied to ROMs, these principles aim to recover discarded mode effects on retained subspace dynamics. The required time-delayed observations and complex, nonlinear transformations suit deep learning approaches

well. Two notable implementations are LSTM-ROM [22] and CDROM [23]. LSTM-ROMs use Long Short-Term Memory networks to correct ROM errors, whilst CDROM employs convolution in time with learned exponential kernels to create memory terms capturing discarded mode effects. CDROM offers improved interpretability and leverages neural ODEs for time continuity. Another pragmatic approach to incorporating memory effects in reduced models is reservoir computing. Fixed recurrent architectures—conceptually related to RNNs, with some analogies to Koopman-inspired embeddings—capture memory effects. Importantly, memory effects are obtained without training internal reservoir dynamics, as only the decoding map is learned. Reservoir computing models are commonly deployed on low-dimensional latent spaces, such as those obtained via proper orthogonal decomposition in fluid dynamics applications [27], since direct application to the full state is typically impractical, and reservoir dimension commonly exceeds input dimension. However, these approaches face challenges. The Takens theorem generally requires significantly more time-delayed observations than system dimensionality, potentially leading to inefficient recovery of discarded dynamics. Furthermore, initialisation poses difficulties: simulations require not only initial conditions but also a sequence of projected dynamics from a full-order model to initialise memory terms. The optimal sequence length is often unclear and depends on the learned time-dependent scheme, complicating initialisation for both LSTM-ROMs and CDROM. We propose an alternative to relying on complex time-dependent dynamics and memory terms. We directly encode discarded modes using more powerful autoencoders, representing unretained information without requiring time-delayed observations. This approach adheres more closely to ROM methodology, with encoding and decoding occurring simultaneously with traditional ROM processes.

Slaved modes hypothesis for closure modelling. A third approach, introduced by Barnett et al. [28], employs a two-tier POD strategy for closure modelling. Known as Neural-Network-Augmented Projection-Based Model Order Reduction (PROM-ANN), this method splits the leading POD modes into two sets: the first contains the most energetic modes, while the second includes lower-energy modes generally discarded in ROMs but believed to significantly influence the dynamics of the first set. The approach assumes that lower-energy mode dynamics are slaved to dominant ones, with a neural network modelling a nonlinear mapping from dominant to lower-energy mode amplitudes. This relation is used to reconstruct the lower-energy modes, and the

coupling term modelling the effects of the lower energy modes on more energetic ones is done via physics based hyper reduction. However, These slave-mode approaches assume the system's full dynamics lie on a manifold fully parametrised by the leading POD coefficients, so that projection onto these modes uniquely determines the evolution of the higher-order modes. This is a strong, often non-trivial assumption, particularly in flows with complex, multiscale modal interactions. The Mori-Zwanzig formalism [21] circumvents such assumptions by showing that the influence of discarded modes can be captured via an additional source term involving time-delayed dominant (retained) modes [21, 23, 22, 29]. While the slaved-mode framework offers a cost-effective solution—especially when aided by neural networks—its general applicability remains uncertain. Our method adopts a two-tier POD strategy similar to PROM-ANN, but crucially does not rely on the slaved-mode hypothesis. Rather than inferring discarded modes from retained ones, we encode and model the dynamics of typically discarded information directly via a neural network. This allows us to represent subspace dynamics explicitly, without assuming functional dependence on retained modes. Another distinction is that PROM-ANN continues to predict dynamics within a truncated POD space after reconstructing lower-energy modes. As demonstrated in Section 7, this approach can be inaccurate for certain problems, whereas our method mitigates truncation limitations through learned closure.

2.2. Methods using Auto-Encoders for dimensionality reduction

Auto-encoder-based methods offer an alternative to POD for dimensionality reduction in ROMs. The nonlinear encoding allows better compression of system dynamics, as illustrated in Figure 1. Convolutional autoencoders further improve representation capacity, leveraging their proven parameter efficiency and effectiveness from computer vision literature [30]. When properly trained, the reduced state can potentially capture all relevant physics, eliminating the need for explicit closure modelling. However, nonlinear learned encodings are generally incompatible with classical projection-based reduced-order models such as POD-Galerkin. This limitation prompts alternative physics-informed approaches to model reduced space dynamics. Examples include Least-Squares Petrov–Galerkin (LSPG) formulations [31, 32, 33] and PDE-constrained optimization techniques like Physics-Informed Neural Networks (PINNs) [34]. While such methods can handle nonlinear latent representations, they are generally more complex and computationally intensive. In contrast, projection-based ROMs—despite potentially costly offline assem-

bly—offer efficient, interpretable, and training-free online evaluation through standard linear algebra. Two main strategies have emerged to address this challenge:

Non-intrusive relearning of dynamics. Fully data-driven ROMs bypass projection entirely and learn latent dynamics from data. Various autoencoder architectures have been proposed, including Multi layer Perceptron autoencoders [35], convolutional autoencoders [36, 37, 38], and attention-based models combined with recurrent networks like ConvLSTM [39], aiming to capture complex spatial and temporal patterns. To address modelling high-dimensional 2D or 3D data using simple MLPs, [40] introduces POD-based preprocessing that retains numerous modes before feeding the reduced representation to an MLP autoencoder. This enables compact latent encodings whilst simplifying the learning task. While these approaches differ in architecture and preprocessing, they share a common trait: latent dynamics must be fully relearned, with no physical structure preserved. Consequently, they may suffer from poor generalisation or violate physical principles like energy conservation. In contrast, our method blends physics-based structure with deep learning flexibility: we retain large-scale POD modes and apply an autoencoder only to intermediate-energy level modes. This limits relearning to a subset of dynamics whilst maintaining interpretability and ensuring accurate reconstructions in physically structured latent spaces.

Physics-based manifold learning. The method introduced by Lee and Carlberg [31] first pre-trains an autoencoder on available data, then applies a Least-Squares Petrov-Galerkin-inspired approach to derive a physics-based model. This model minimises the residual to identify a latent vector that best satisfies the governing equations. However, it typically requires residual evaluation in the full physical space, which can undermine the computational benefits of model reduction. To address this, hyper-reduction techniques decode only at selected spatial locations rather than the entire domain. While this lowers computational cost, it introduces approximation errors in the residual, as examined in studies of hyper-reduction methods such as DEIM and ECSW [41, 42, 43], which analyse trade-offs between accuracy and efficiency in reduced-order models. Our method differs by employing a hybrid approach that jointly learns dynamics estimation with the encoding and decoding processes. This promotes consistency between the smoothness and regularity of the encoding and the model's representativity and reconstruction capability. Moreover, our method avoids decoding in the full space or

relying on hyper-reduction at each timestep for dynamics estimation, thereby eliminating the associated computational overhead and approximation errors.

Model	Projection method	Modelling method	Time continuous	Time delayed observations
LSTM-ROM	Linear	Learnt	No	Yes
CDROM	Linear	Hybrid	Yes	Yes
PROM-ANN	Linear	Physics-based	Yes	No
Manifold Learning	Nonlinear	Physics-based	Yes	No
CAE-GPR ROM	Nonlinear	Learnt	Yes	No
Lusch and al.	Nonlinear	Learnt	Yes	No
Our Approach	Hybrid	Hybrid	Yes	No

Table 1: Key aspects of existing approaches vs. our new hybrid model.

3. Background

3.1. Scientific Context and Problem Statement

Throughout, UPPERCASE denote matrices, **bold** lowercase denote vectors, and regular lowercase denote scalars.

We consider nonlinear partial differential equations (PDEs) discretised by finite elements in the semi-discrete weak form (1):

$$M d_t \mathbf{w} = \mathcal{L}\mathbf{w} + \mathcal{B}(\mathbf{w}, \mathbf{w}) \tag{1}$$

Here M is the mass matrix, and $\mathbf{w}$ a high-dimensional state vector. We assume $\mathbf{w}$ is homogeneous with respect to all boundary conditions, i.e., if $\mathbf{w}$ satisfies them, then so does $\lambda\mathbf{w}$ for any scalar λ. The dynamics involve two weak-form operators: $\mathcal{L}$ (linear) and $\mathcal{B}$ (bilinear). Whilst the proposed method applies to arbitrary nonlinearities $\mathcal{N}(w)$, we focus on problems governed by quadratic nonlinearities. This form covers many fluid-mechanics equations, including incompressible Navier-Stokes equations in perturbative form (e.g., with respect to the base flow), simplified 1D models like viscous Burgers equations, and certain compressible Navier-Stokes problems (under suitable variable changes) [44]. Moreover, many nonlinear PDEs encountered in engineering can be lifted to quadratic form through variable transformations [45], making this formulation broadly applicable.

Although classical methods can resolve such systems, our goal is to compute new solutions efficiently—under changes in initial condition $\mathbf{w}(t = 0)$

or parametric variations in $\mathcal{L}$ and $\mathcal{B}$. The challenge is developing ROMs that accurately capture the system's behaviour and solution space whilst substantially reducing computational cost. Although this work focuses on bilinear weak formulations, the proposed approach can be extended to more general nonlinearities using hyper-reduction techniques. This perspective is discussed in section 9, where we outline how such extensions could support more complex systems whilst maintaining computational efficiency.

3.2. POD-based ROM methodology

We construct the ROM about a steady base flow. Let $\mathbf{w} = \mathbf{w}_0 + \mathbf{w}'$ where $\mathbf{w}_0$ is the base flow (solution of the stationary Navier–Stokes equations for the chosen parametrisation and problem) and $\mathbf{w}'$ the perturbation. In weak form, the base flow satisfies $\mathcal{L}\mathbf{w}_0 + \mathcal{B}(\mathbf{w}_0, \mathbf{w}_0) = \mathbf{0}$. Substituting into (1) and using this relation gives:

$$M d_t \mathbf{w}' = \mathcal{L}'\mathbf{w}' + \mathcal{B}(\mathbf{w}', \mathbf{w}'), \tag{2}$$

with $\mathcal{L}'\mathbf{w}' = \mathcal{L}\mathbf{w}' + \mathcal{B}(\mathbf{w}_0, \mathbf{w}') + \mathcal{B}(\mathbf{w}', \mathbf{w}_0)$ the linear part of the governing operator. The ROM is derived from equation (2) and in the following, $\mathbf{w}'$ will be noted $\mathbf{w}$ for brevity. This choice affects the parametric formulation discussed in 8, since $\mathbf{w}_0$ depends nonlinearly on the parameter.

The ROM is obtained by projecting the governing equations onto a well-chosen reduced orthogonal basis. Such a basis is obtained using POD by stacking snapshots of the state vector w' into a data matrix X. The spatial and temporal POD modes relate to the left and right singular vectors of X, respectively, ranked by decreasing singular value. In practice, they are obtained by first computing the covariance matrix C, defined as:

$$C = X^\top M X \tag{3}$$

where we have assumed that the mass-matrix M is positive, symmetric and corresponds to a relevant correlation measure between snapshots. An eigen-decomposition of C then yields the matrix Ψ containing all temporal POD modes [15]:

$$C\Psi = \Psi\Sigma \tag{4}$$

with Σ the diagonal matrix containing the eigenvalues associated with each eigenvector (each column) of Ψ, ranked by decreasing order [15]. The spatial POD modes are then given by

$$\Phi = X\Psi\Sigma^{-\frac{1}{2}} \tag{5}$$

The spatial POD modes are orthonormal: $\Phi^\top M \Phi = I$.

We obtain the following relation, with $\boldsymbol{\varphi_i}$ the ith column of matrix Φ, $\mathbf{a}$ the vector containing all POD coefficients and a_i the coefficient of the i^{th} POD mode:

$$\mathbf{w}' = \Phi \mathbf{a} = \sum_{i=0}^{N_t - 1} \boldsymbol{\varphi_i} a_i \tag{6}$$

To form the reduced basis, eigenvectors associated with the r-largest eigenvalues (4) are selected, as they form the r-dimensional linear subspace capturing most of the snapshot variance. This yields a truncated eigenvector matrix Ψ_r and eigenvalues matrix Σ_r. The truncated spatial POD basis is then given by equation (5) as the columns of $\Phi_r = X \Psi_r \Sigma_r^{-\frac{1}{2}}$.

The governing equations of the Full Order Model (FOM) are projected onto Φ_r, yielding the following reduced PDE:

$$d_t \mathbf{a} = \Phi_r^\top \mathcal{L}' \Phi_r \mathbf{a} + \Phi_r^\top \mathcal{B}(\Phi_r \mathbf{a}, \Phi_r \mathbf{a}) \tag{7}$$

with $\Phi_r^\top M \mathbf{w} = \mathbf{a}$ the projection of a snapshot from the physical space to the reduced basis. We can then compute the ROM as:

$$d_t a_i = \mathrm{ROM}_i(\mathbf{a}) \tag{8}$$

where

$$\begin{aligned} \mathrm{ROM}_i(\mathbf{a}) =& \sum_{j=0}^{r-1} L_{ij} a_j + \sum_{j,k=0}^{r-1} N_{ijk} a_j a_z \\ L_{ij} =& \boldsymbol{\varphi}_i^\top \mathcal{L}' \boldsymbol{\varphi}_j, \; N_{ijk} = \boldsymbol{\varphi}_i^\top \mathcal{B}(\boldsymbol{\varphi}_j, \boldsymbol{\varphi}_k) \end{aligned} \tag{9}$$

3.3. Projection and reduced trajectory error

The discrepancy between a Galerkin-POD-based ROM and the Full order model (FOM) is due to two sources of error: the projection error and the reduced trajectory error. Truncating the spatial POD mode matrix leaves a residual term $\widetilde{\mathbf{w}}$ that reads:

$$\widetilde{\mathbf{w}} = \mathbf{w} - \mathbf{w}_{\mathrm{POD}}, \text{ with } \mathbf{w}_{\mathrm{POD}} = (\mathbf{w_0}) + \Phi_r \Phi_r^\top M (\mathbf{w} - \mathbf{w_0}) \tag{10}$$

The truncation error stems from the fact that the FOM solution evolves in a space wider than the reduced linear subspace associated with $\mathbf{w}_{\mathrm{POD}}$. In addition to this error, the unrepresented modes affect the time evolution of

$\mathbf{w}_{\text{POD}}$ due to the coupled nature between $\widetilde{\mathbf{w}}$ and the latter. In particular, the term $\Phi_r^\top \mathcal{L}' \widetilde{\mathbf{w}}$ is generally non-zero and the projected nonlinear term may be expressed by decomposing $\Phi_r^\top \mathcal{B}(\mathbf{w}, \mathbf{w})$ as:

$$\begin{aligned} \Phi_r^\top \mathcal{B}(\mathbf{w}, \mathbf{w}) =& \Phi_r^\top \mathcal{B}(\mathbf{w}_{\text{POD}}, \mathbf{w}_{\text{POD}}) + \Phi_r^\top \mathcal{B}(\widetilde{\mathbf{w}}, \widetilde{\mathbf{w}}) \\ &+ \Phi_r^\top \mathcal{B}(\mathbf{w}_{\text{POD}}, \widetilde{\mathbf{w}}) + \Phi_r^\top \mathcal{B}(\widetilde{\mathbf{w}}, \mathbf{w}_{\text{POD}}) \end{aligned} \tag{11}$$

Only the first term of the right-hand side of equation (11) is accounted for in the POD-based ROM dynamics whilst the others are neglected. These approximations only hold if the neglected terms are indeed weak, which is true only if the amplitude decay in a_i is sufficiently strong with respect to coefficients arising in $\mathcal{L}'$ and $\mathcal{B}$ (which cannot be guaranteed). Therefore, not accounting for $\widetilde{\mathbf{w}}$ leads to an erroneous trajectory in the reduced linear subspace, with an error on $\mathbf{w}_{\text{POD}}$ that accumulates over time. We introduce a new method where a nonlinear autoencoder is used to reduce both the truncation and the reduced trajectory error (whilst other recent similar works such as [23] only focus on the reduced trajectory error).

4. Proposed Approach

4.1. AE-augmented POD-based method

Our goal is to retain ROM efficiency whilst recovering FOM accuracy. Rather than relying solely on the POD subspace, we augment it with an autoencoder to recover dynamics commonly lost through truncation. The novelty lies in hybridising classical POD with deep learning, balancing physical interpretability with data-driven expressive power.

The model comprises three components: a hybrid POD–autoencoder encoder, a hybrid time integration module, and a hybrid decoder (Figure 2). This structure merges POD and deep learning strengths throughout the modelling pipeline.

Dimensionality reduction follows a two-tier strategy. The most energetic modes are preserved using classical POD to ensure interpretability and enable physics projection. A number of less energetic modes are nonlinearly encoded using an autoencoder into a compact latent representation. This hybrid latent space enables accurate reconstruction of fine-scale features with minimal loss.

Time integration combines a POD-Galerkin ROM with deep learning. The POD component predicts coarse dynamics, whilst a feedforward neural

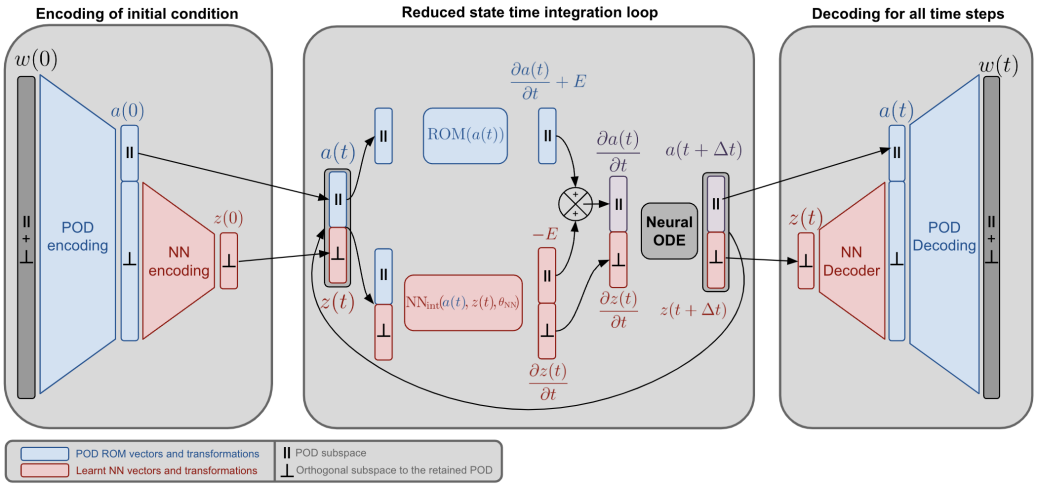


Figure 2: Proposed hybrid model architecture. In the reduced basis projection, we use traditional POD augmented with extra modes from nonlinear encoding of unretained POD information. Time integration employs a hybrid neural ODE, combining POD-based ROM output with neural network corrections. Finally, in decoding, time-integrated reduced states are mapped to the physical state using both POD mode representation and nonlinear decoding of unretained information.

network corrects errors in the POD subspace and learns full dynamics in the AE subspace. This hybrid derivative estimation, shown in the centre block of Figure 2, fits naturally within the Neural ODE framework [46], providing continuous and stable time-stepping aligned with physical systems.

The right block of Figure 2 depicts the hybrid decoder, which reconstructs physical states at each time step. It preserves retained POD modes and maps the AE latent space to second-tier POD modes. Due to POD orthogonality, only the AE subspace corrects information beyond retained linear modes.

4.2. Model architecture

This section details the three components outlined above. Figure 2 provides a visual overview of the model structure, whilst Algorithm 1 offers a detailed, step-by-step description of the method and training process.

Encoder. The encoder transforms the state vector $\mathbf{w}(t)$ into a reduced state $\big(\mathbf{a}_{\parallel}(t), \mathbf{z}(t)\big)$ following (time dependence omitted to simplify notation):

$$\begin{aligned} \mathbf{a}_{\parallel}(t) &= \Phi_{\parallel}^{\top} M \mathbf{w}(t) \\ \mathbf{z}(t) &= \text{Encoder}(\mathbf{a}_{\perp}(t), \theta_E), \quad \mathbf{a}_{\perp}(t) = \Phi_{\perp}^{\top} M \mathbf{w}(t) \end{aligned} \tag{12}$$

This reduction uses a two-tier POD strategy:

- The first tier POD, $\Phi_{\|}$ in equation (12), retains only the most energetic POD modes. Coefficients $\mathbf{a}_{\|}(t)$ form an orthonormal basis representing a reduced subspace for Galerkin projection.

- The second tier, $\Phi_{\perp}$ in equation (12), retains a large number of lower-rank modes (but not all). These modes will not be modelled physically. Coefficients $\mathbf{a}_{\perp}(t)$ are further compressed using neural network encoder $\text{Encoder}(\mathbf{a}_{\perp}, \theta_E)$, resulting in substate z containing information absent from the first-tier POD subspace.

This approach provides accurate yet simple, physics-based representation of energetic large-scale structures. Given fixed computational budget, we adopt pragmatic two-tier(+discarded) decomposition. The first tier contains large-scale modes of primary physical interest. The second tier gathers POD modes that interact most with the first tier and are necessary for physics reconstruction; these often lack interpretability and can be numerous, so our method represents them compactly via nonlinear compression, with coefficients $a_{\perp}$ in the M-orthogonal complement of the first-tier subspace. The discarded set comprises remaining modes—very low-energy content not contributing meaningfully to targeted quantities—so concatenation $[a_{\|}; a_{\perp}]$ defines a reduced state smaller than full-order state $\mathbf{w}$. This organisation preserves the original trade-off: we prioritise traditional linear POD for leading dynamics and interpretability, but allocate remaining budget to nonlinear encoders that better compress residual interactions and small-scale content. The encoder method is described in lines 9–11 of Algorithm 1.

The two-tier POD approach offers key advantages over using nonlinear encoders on raw physical space data:

- The neural network encoder size is independent of the physical state vector dimension, scaling instead with POD mode truncation level, offering favourable computational scaling for larger systems.

- Original simulation data on complex unstructured 2D meshes poses challenges for traditional autoencoders. Whilst Graph Neural Networks handle such data, their complexity makes implementation challenging. POD encoding transforms mesh data into simplified one-dimensional vectors, making simple multilayer perceptron architectures suitable.

- POD preprocessing acts as an adjustable spatial frequency filter, capturing dominant flow structures whilst reducing small-scale fluctuation influence. This retains qualitatively relevant information and could benefit noisy or turbulent data applications, though not explored here.

- Empirical results indicate these advantages lead to improved model smoothness and accuracy.

Time Integration. We employ neural ODE framework for time integration. The deep learning problem is formulated as a differential equation, where $f(\mathbf{w}(t), \theta)$ represents a hybrid model and hat-quantities $\widehat{\cdot}$ represent estimated dynamics:

$$d_t \hat{\mathbf{w}}(t) = f(\hat{\mathbf{w}}(t), \theta) \tag{13}$$

To integrate the time derivative from t to $t + \Delta t$, we employ numerical methods:

$$\hat{\mathbf{w}}(t + \Delta t) = \hat{\mathbf{w}}(t) + \int_t^{t+\Delta t} f(\hat{\mathbf{w}}(\tau), \theta) d\tau \tag{14}$$

The reduced state derivative is computed differently for each subspace (see line 12 of Algorithm 1). In the POD subspace, the ROM estimates $d_t \mathbf{a}_\parallel$ using the POD–Galerkin model (Section 3.2). Simultaneously, neural network $\mathrm{NN}_\parallel$ acts as closure model, correcting this estimate using inputs from both POD and autoencoder subspaces for comprehensive dynamics correction. In the orthogonal autoencoder subspace, dynamics are entirely neural network-driven. Network $\mathrm{NN}_\perp$ computes $d_t \mathbf{z}$ using information from both subspaces, enabling nonlinear interactions. Although lowest-energy modes are discarded, residual influence on retained dynamics can be learned implicitly from data by the end-to-end model. Complete derivative estimation is:

$$\begin{aligned} d_t \hat{\mathbf{a}}_\parallel(t) &= \mathrm{ROM}(\hat{\mathbf{a}}_\parallel(t)) + \mathrm{NN}_\parallel(\hat{\mathbf{a}}_\parallel(t), \hat{\mathbf{z}}(t), \theta_{NN}) \\ d_t \hat{\mathbf{z}}(t) &= \mathrm{NN}_\perp(\hat{\mathbf{a}}_\parallel(t), \hat{\mathbf{z}}(t), \theta_{NN}) \end{aligned} \tag{15}$$

where $\mathrm{ROM}(\hat{\mathbf{a}}_\parallel(t))$ is defined in (9), $\mathrm{NN}_\parallel(\cdot)$ and $\mathrm{NN}_\perp(\cdot)$ denote network outputs (correction for $d_t \hat{\mathbf{a}}_\parallel$ and estimate of $d_t \hat{\mathbf{z}}$, respectively). Term $\mathrm{ROM}(\hat{\mathbf{a}}_\parallel(t))$ accounts for autonomous dynamics within the POD subspace. Term $\mathrm{NN}_\parallel(\hat{\mathbf{a}}_\parallel(t), \hat{\mathbf{z}}(t), \theta_{NN})$ captures coupling dynamics between POD and autoencoder subspaces. Finally, $\mathrm{NN}_\perp(\hat{\mathbf{a}}_\parallel(t), \hat{\mathbf{z}}(t), \theta_{NN})$ represents full orthogonal subspace dynamics, including autonomous and coupling effects. This hybrid approach leverages POD-Galerkin interpretability whilst incorporating

neural network flexibility. Our model estimates all three coupling terms outlined in section 3.3, often neglected in traditional ROMs or only partially considered by methods working solely in POD subspace [22, 23, 28]. With derivative estimation explained, time-integration from t to $t+\Delta t$ uses neural ODE method. Initial conditions can be ground-truth quantities $(\mathbf{a}_{\|}(t), \mathbf{z}(t))$ or predicted ones $(\hat{\mathbf{a}}_{\|}(t), \hat{\mathbf{z}}(t))$.

Decoder. After time integration, the predicted full state is recovered following:

$$\hat{\mathbf{w}}(t) = \Phi_{\|}\hat{\mathbf{a}}_{\|}(t) + \Phi_{\perp}\hat{\mathbf{a}}_{\perp}(t), \quad \hat{\mathbf{a}}_{\perp}(t) = \text{ Decoder}(\hat{\mathbf{z}}(t), \theta_D) \tag{16}$$

Autoencoder latent variable dynamics are first decoded using neural network $\text{Decoder}(\hat{\mathbf{z}}(t), \theta_D)$ to recover $\hat{\mathbf{a}}_{\perp}$. First and second-tier coefficients are projected back to full physical space using POD projection matrix. This hybrid decoding enforces orthogonality as latent variable decoding remains orthogonal to first-tier POD modes. The decoder maintains POD orthogonality by keeping predicted retained POD modes unchanged whilst the neural network decoder affects only second-tier POD modes. Due to orthogonality, the decoder cannot output information in the first-tier POD subspace. Consequently, perfect reconstruction requires accurate prediction of both first-tier and second-tier POD mode dynamics. The decoder architecture is a multilayer perceptron, typically with symmetric structure to the encoder. In some variants, the decoder contains additional parameters (e.g., one extra layer) to enhance expressiveness, reflecting the greater complexity of decoding tasks where more expressive decoders frequently improve performance.

4.3. Model training

In the previous section, we outlined the model architecture. However, in deep learning, architectural design and function approximation capabilities represent only part of the task. Training the model—i.e., tuning the weights to achieve the desired outcome—requires numerous decisions. This section outlines those choices.

Learning problem formulation. The learning task is defined by input-output pairs, where the input is the initial condition and the output is the corresponding trajectory. Specifically, we consider ground truth trajectories $\mathbf{w}(t)$, post-processed using our two-tier POD decomposition to obtain time-dependent coefficients $\mathbf{a}_{\|}(t)$ and $\mathbf{a}_{\perp}(t)$. The model input, $\big(\mathbf{a}_{\|}(0), \mathbf{a}_{\perp}(0)\big)$, represents the POD coefficients of the initial condition. The model output is the

Algorithm 1: Hybrid model and training

Initialisations

Load M, A, N `/* Finite element mass matrix and operators`

Load $\Phi_{\|}, \Phi_{\perp}$ `/* two tier POD modes matrices`

Load ROM() with $(\Phi_{\|}, A, N)$ `/* initialise the ROM`

Initialise NN() `/* time integration network`

Initialise Encoder() `/* encoder network to encode` $\mathbf{a}_{\perp}^{t}$

Initialise Decoder() `/* decoder network to decode` $\mathbf{z}^{t}$

for $epoch = 0 : N_{epochs}$ **do**

  $Loss = 0$ `/* Reinitialise Loss`

  for $\mathbf{w}_m(0:T),\ m = 0 : N_{examples}$ **do**

    # Encoding #

    $\mathbf{a}_{\|}(0:T) = \Phi_{\|}^{\top} M \mathbf{w}(0:T)$ `/* tier 1 POD Reduction`

    $\mathbf{a}_{\perp}(0:T) = \Phi_{\perp}^{\top} M \mathbf{w}(0:T)$ `/* tier 2 POD Reduction`

    $\mathbf{z}(0:T) = \text{Encoder}(\mathbf{a}_{\perp}(0:T), \theta_E)$ `/* non linear reduction`

    # Time integration #

    $$\begin{bmatrix} \hat{\mathbf{a}}_{\|}(0:T) \\ \hat{\mathbf{z}}(0:T) \end{bmatrix} = \begin{bmatrix} \mathbf{a}_{\|}(0) + \int_0^T \left[\text{ROM}(\hat{\mathbf{a}}_{\|}(t)) + \text{NN}_{\|}(\hat{\mathbf{a}}_{\|}(t), \hat{\mathbf{z}}(t), \theta_{NN})\right] dt \\ \mathbf{z}(0) + \int_0^T \text{NN}_{\perp}(\hat{\mathbf{a}}_{\|}(t), \hat{\mathbf{z}}(t), \theta_{NN})\, dt \end{bmatrix}$$

    # Decoding #

    $\hat{\mathbf{a}}_{\perp}(0:T) = \text{Decoder}(\hat{\mathbf{z}}(0:T), \theta_D)$

    $\hat{\mathbf{w}}(0:T) = \Phi_{\|} \hat{\mathbf{a}}_{\|}(0:T) + \Phi_{\perp} \hat{\mathbf{a}}_{\perp}(0:T)$

    # Loss and back propagation #

    $$\begin{aligned} Loss = & L_{\text{recon}}(\hat{\mathbf{a}}_{\|}(0:T), \mathbf{a}_{\|}(0:T), \hat{\mathbf{a}}_{\perp}(0:T), \mathbf{a}_{\|}(0:T)) \\ & + \lambda L_{\text{regularisation}}(\hat{\mathbf{z}}(0:T), \mathbf{z}(0:T)) \end{aligned}$$

  end

  $\theta_{NN} = \theta_{NN} - \alpha \frac{\partial \mathbf{Loss}}{\partial \theta_{\mathbf{NN}}}$ `/* Back propagation and weight update`

  $\theta_E = \theta_E - \alpha \frac{\partial \mathbf{Loss}}{\partial \theta_{\mathbf{E}}}$

  $\theta_D = \theta_D - \alpha \frac{\partial \mathbf{Loss}}{\partial \theta_{\mathbf{D}}}$

end

predicted POD coefficients at subsequent time steps, denoted $\left(\hat{\mathbf{a}}_{\|}(t), \hat{\mathbf{a}}_{\perp}(t)\right)$

for $t > 0$. This defines learning pairs $\left\{ \left(\mathbf{a}_{\parallel}(0), \mathbf{a}_{\perp}(0)\right), \left(\mathbf{a}_{\parallel}(t), \mathbf{a}_{\perp}(t)\right)_{t>0} \right\}$, with the loss formulated to minimise the discrepancy between predicted and true coefficients, as described in the next section.

Reconstruction Loss Formulation. The model's modular architecture supports multiple loss formulations, enabled by its partially physics-based design. We begin with the reconstruction loss, which trains the model to reproduce ground truth trajectories. Three formulations were tested.

First, we used a loss on the POD coefficients, comparing the two-tier POD encoding of the predicted and reference trajectories. This directly measures the discrepancy between predicted and true POD coefficients.

Second, we assessed a loss in the full physical space, computed after full decoding. This more closely reflects the goal of accurate reconstruction in the original domain.

Third, we tested a physics-informed loss using the H^1 norm, which incorporates spatial gradients and is natural in finite element contexts.

These losses increase in complexity, potentially improving physical accuracy. However, added transformations (particularly on long rollout trajectories) complicate the loss landscape. A balance is required between physical fidelity and ease of optimisation. Hyperparameter tuning showed that the H^1-based loss was the most unstable, frequently causing exploding gradients or convergence to poor local minima. Similarly, losses in full physical space often led to stagnation, especially for the cylinder flow case, where the encoder-decoder collapsed to near-zero outputs. Due to consistent training failures, these settings are excluded from the reported results. Empirically, we adopt the POD coefficient-based loss for all models. Theoretically, this choice minimises transformations from target to prediction while remaining physically interpretable. As POD modes are normalised, dominant modes yield larger coefficients. Computing the mean squared error (MSE) on these coefficients naturally emphasises dominant dynamics, aligning with our modelling goals. Section 8.3 further discusses loss normalisation to address varying trajectory norms in the parametric setting.

The final reconstruction loss is defined as:

$$L_{\text{recon}} = \frac{1}{N_t} \sum_{n=1}^{N_t} \left(\|\hat{\mathbf{a}}_{\parallel}(n\Delta t) - \mathbf{a}_{\parallel}(n\Delta t)\|^2 + \|\hat{\mathbf{a}}_{\perp}(n\Delta t) - \mathbf{a}_{\perp}(n\Delta t)\|^2 \right) \quad (17)$$

where Δt is the sampling interval. For clarity, we assume that loss evaluation and simulation share the same time step, although the neural ODE

framework supports arbitrary sampling intervals.

Deep Supervision Loss. In addition to the reconstruction loss, we introduce a regularisation term applied to the autoencoder's latent state:

$$L_{\text{regularisation}} = \frac{1}{N_t} \sum_{n=1}^{N_t} \|\hat{\mathbf{z}}(n\Delta t) - \mathbf{z}(n\Delta t)\|^2 \tag{18}$$

This term computes the MSE between the encoded ground-truth snapshots, $\mathbf{z}(t)$, and the corresponding time-integrated latent trajectory, $\hat{\mathbf{z}}(t)$, obtained from the initial condition.

It serves a dual purpose. First, it ensures the autoencoder and neural time integrator operate within a shared latent space, preventing the encoder from overfitting to training initial conditions. Second, it provides deep supervision [47], improving gradient flow to the encoder. Given the model's autoregressive nature, gradients become increasingly complex across multiple time steps.

We did not perform an ablation study on this term, as it seems essential in preventing overfitting to training initial conditions. The full loss combines both terms, with a hyperparameter λ controlling their relative weights. In practice, we set $\lambda = 1$. The complete training loss is:

$$L_{\text{full}} = L_{\text{recon}} + \lambda L_{\text{regularisation}} \tag{19}$$

Simultaneous training of all neural networks. The model architecture, comprising multiple neural network components, allows for flexible training strategies. This versatility enables us to tailor the learning process to distinct parts of the model. While one could pre-train the encoder and decoder separately—ensuring accurate encoding of POD-discarded information before training the time integrator—we chose to train all neural networks jointly (with the POD basis pre-trained and fixed). This choice was motivated by potential drawbacks of unregularised autoencoders, which may yield overly complex latent spaces and hinder learning of the associated dynamics. Joint training promotes a balanced trade-off between latent space expressivity and regularity, allowing the time integration network to implicitly regularise the latent space.

Length of Training Trajectories. The training procedure follows the neural ODE framework, computing the loss over time-integrated trajectories. This

enhances stability and accuracy, particularly during autoregressive inference. However, determining an appropriate rollout length during training remains an open issue in neural ODE methods [37, 48]. Training on long trajectories from the outset often results in unstable predictions, especially when the network is randomly initialised and coupled with a ROM. To address this, we initially train on truncated trajectories and gradually increase the integration length. This warm-up strategy reduces divergence risks in the early training stages.

Neural ODE Implementation. Although the neural ODE framework enables higher-order time integration than the Euler method, experiments indicate that the integration scheme has limited impact on accuracy when the time step is kept consistent. This is likely because the neural network learns to correct numerical errors introduced by the integration scheme. We therefore adopt the Euler method for time integration, balancing accuracy and numerical efficiency while simplifying ROM integration via constant time steps. Neural ODEs often use adjoint equations for backpropagation to achieve constant memory complexity. However, these methods can be inaccurate in sensitivity estimation and backwards integration for dissipative systems, resulting in inaccurate gradients. Techniques such as Adaptive Checkpoint Adjoint [49] attempt to mitigate these issues, but in our tests, automatic differentiation through the Euler computation graph performed the best. While automatic differentiation may suffer from vanishing or exploding gradients in deep computation graphs, and memory complexity grows with network size, our hybrid architecture avoids these drawbacks. With part of the computation handled by the ROM, the neural network remains compact, reducing backpropagation cost and complexity. We therefore employ a simple neural ODE scheme (Euler integration with non-adjoint backpropagation). This choice balances accuracy, efficiency, and architectural coherence. Although further tuning of the neural ODE component might yield marginal gains, such optimisation falls outside the scope of this study, which centres on the ROM–neural network hybrid.

Scope of the comparison to other models. Intrusive reduced-order models are highly sensitive to discretisation choices and implementation details. Therefore all baselines were re-implemented within our framework and tuned with the same hyper-parameter search (see Appendix A for code, settings, and training-time statistics). We benchmark our hybrid approach against four reference models that collectively cover the main ROM families:

- Fully data-driven model: a neural-ODE model that uses the same large-scale POD projection, followed by autoencoder compression of all modes. This isolates the benefit of hybridisation and represents two-stage approaches such as [26], while remaining implementation-wise as close as possible to our method.
- POD-Galerkin ROM: the canonical intrusive baseline for linear projection methods.
- POD truncation error: the theoretical lower bound for any model confined to the retained POD subspace, encompassing closure strategies such as CD-ROM, VMS-ROM, and LSTM-POD-ROM.
- Large-scale POD-Galerkin ROM (pinball only): all first and second tier modes are included in the Galerkin projection, as in Section 7. This represents a best-case scenario for projection–hybrid methods (e.g. PROM-ANN [28]), yet demonstrates that even with high energy retention and many modes, truncation–projection alone cannot ensure accurate dynamics.

This controlled four-way comparison clarifies where the proposed hybrid model adds value relative to purely data-driven, purely projection-based, and best-case projection-hybrid approaches.

5. Viscous Burgers' equation results

5.1. Test case description and discussion

We consider the one-dimensional (1D) viscous Burgers' problem as a test case, governed by the following partial differential equation (PDE) :

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} - \nu\frac{\partial^2 u}{\partial x^2} = 0, \qquad x \in (0,1),\ t > 0,$$

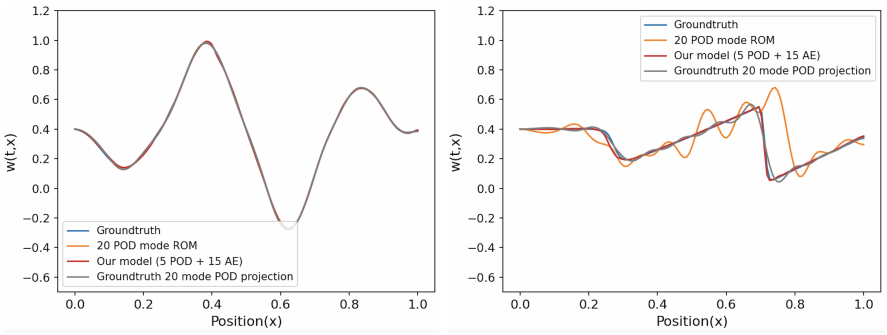
where u is the 1D velocity field and ν a positive constant. This case combines low numerical cost with highly convective and nonlinear behaviour. We intentionally choose small viscosity ν values to emphasise this regime, increasing relative convection and promoting the formation of sharp travelling fronts. These features resemble high-Reynolds-number flows [50, 51, 52] and pose additional challenges for model reduction, beyond convection and nonlinearity. POD-Galerkin reduced-order models (ROMs) face two main

difficulties in this context: (i) a linear projection basis cannot efficiently capture the steep, moving gradients of sharp fronts, requiring an impractically large number of modes; and (ii) the predominantly nonlinear dynamics remain challenging to model, even with many retained modes. This makes the Burgers' problem a relevant test case for assessing our model's capabilities.

Thanks to the reduced computational cost (fewer degrees of freedom and time steps) we conduct numerous experiments on this case, enabling detailed performance comparisons and rapid iteration before tackling more complex scenarios.

We use a first-order finite-element method (FEM) for spatial discretisation and an Euler scheme for time integration, with the viscosity term treated implicitly and the convection term explicitly. Simulations are run with FEniCSx [53, 54] on a 128-element unit-length mesh and time step $\Delta t = 0.005$, with constant viscosity $\nu = 10^{-3}$. Initial conditions are sampled from a Perlin-noise distribution [55] with unit wavelength and a random number of harmonics in the range $(1, 6)$, centred at $u = 0.4$.

5.2. General comparative results

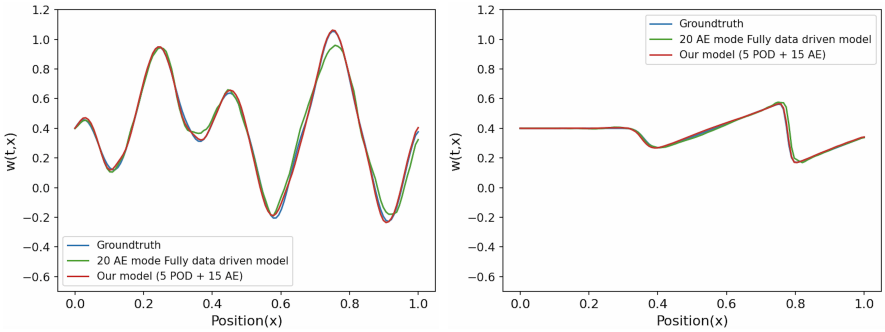


(a) Initial condition

(b) Snapshot at 150 timesteps

Figure 3: Visualisation of 3 models performance on a specific example at 3 different time steps. The compared models are our model with a latent space composition of 5 POD modes and 15 AE variables, The projection of the ground truth in a 20 mode POD subspace (acting as a best case scenario for models working only in the POD subspace) and a 20 mode POD Galerkin ROM.

Qualitative comparison with other models. We perform a qualitative comparison of our hybrid model against both POD-based and fully data-driven autoencoder models, as shown in Figures 3 and 4. For clarity, the comparison is separated between POD-based and autoencoder-based methods.



(a) Initial condition (b) Snapshot at 200 timesteps

Figure 4: Visualisation of 2 models performance on a specific example at 3 different time steps. The compared models are our model with a latent space composition of 5 POD modes and 15 AE variables, and a fully data driven model with 20 AE variables encoding.

All models use a latent space of 20 modes to better visualise and highlight discrepancies. Although larger latent spaces typically enhance performance, the 20-mode setting provides clearer visual distinctions, aiding interpretation. The trends observed at this size are representative of those seen with larger latent spaces and will be confirmed in the remainder of the Burgers' equation results.

In the first comparison, we assess our model against a traditional POD-Galerkin ROM and the projection of the ground truth onto the 20-mode POD subspace, serving as a best-case scenario for POD-only methods. Figure 3 illustrates the challenge posed by the Burgers equation for POD approaches, primarily due to its strong nonlinearity. The traditional ROM fails to capture the correct dynamics and exhibits severe oscillations, while even the best-case POD projection cannot resolve the sharp convective fronts and introduces nonphysical oscillations near the front. Our hybrid model, by combining the ROM structure with autoencoder-based correction, significantly reduces these artefacts and better captures the system's physical behaviour.

In the second comparison (Figure 4), we evaluate our model against a fully data-driven autoencoder using an encode-neural ODE-decode architecture. Although this model outperforms POD-based methods on the Burgers equation, it lacks the physical grounding of ROMs. This limitation manifests in two areas: the initial condition encoding is less accurate, and while the convective fronts are better represented than in POD models, their propagation speed is inaccurate, sometimes advancing too quickly or too slowly,

especially at timestep 200. Our method avoids these issues due to the ROM contribution.

These comparisons are based on our model's two worst-performing test examples, complex scenarios that challenge all methods. Nonetheless, our model demonstrates robust performance, handling even these difficult cases effectively, further supporting its stability and reliability.

Quantitative Comparison with Other Models. To assess overall model performance, we compare the average test set mean square error (MSE) of our hybrid model variants with other approaches. The results are summarised in Table 2.

We evaluate the effect of latent space composition by varying the number of POD and AE latent variables across 3 hybrid model variants: 20 POD and 10 AE variables, 10 POD and 20 AE variables, and 5 POD and 25 AE variables. For comparison, we include three additional models: a fully data-driven autoencoder (AE) with 30 latent variables, a linear hybrid model with 30 POD modes and no AE variables, and a POD-Galerkin ROM with 30 modes. We also report the projection error of a 30-mode POD, which serves as an upper bound for ROMs restricted to the POD subspace with perfect dynamics.

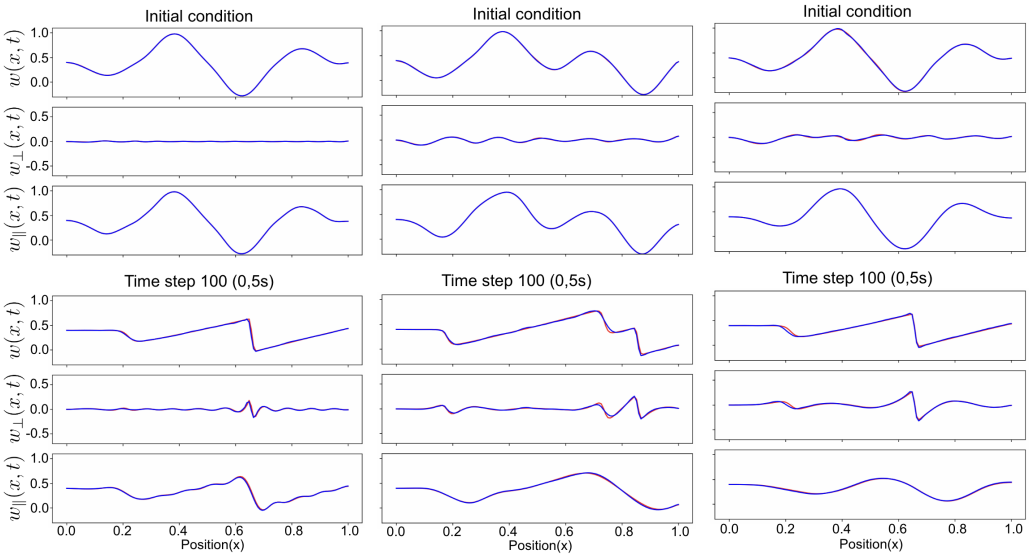
As shown in Table 2, the hybrid model with 20 POD and 10 AE latent variables achieves the lowest test set MSE (8.364×10^{-6}), outperforming the fully data-driven model, the linear hybrid model, and the POD-Galerkin ROM. Remarkably, it also surpasses the 30-mode POD projection error (7.73×10^{-5}), indicating that AE latent variables enhance information density beyond linear modes.

The choice of a 30-mode latent space is motivated by two factors. First, on the full dataset, the POD projection error becomes visually satisfactory at this threshold, with only minor non-physical oscillations. Second, 30 modes represent the performance threshold for ROMs trained on a single trajectory. This provides a practical benchmark: matching single-trajectory ROM performance while accommodating the greater variability of the full dataset.

These results highlight the relevance of optimal latent space composition, as performance varies across hybrid configurations. This notion is further explored in Section 5.3.

Model	Test-set MSE
Hybrid model 20 POD + 10 AE modes	$\mathbf{8.364 \times 10^{-6}}$
Hybrid model 10 POD + 20 AE modes	1.302×10^{-5}
Hybrid model 5 POD + 25 AE modes	2.154×10^{-5}
Linear encoding hybrid model, 30 POD + 0 AE modes	8.265×10^{-5}
Fully data-driven model 30 modes	7.655×10^{-5}
POD-Galerkin ROM 30 modes	1.600×10^{-3}
30 modes POD projection error (best case scenario for POD-based models)	7.73×10^{-5}

Table 2: Test set mean square error (MSE) for hybrid models and comparative models with 30 total modes in the latent space.



(a) Hybrid model : 20 POD + 20 AE modes

(b) Hybrid model : 10 POD + 15 AE modes

(c) Hybrid model : 5 POD + 15 AE modes

Figure 5: Visualisation of prediction of the test set 0.5s (100 time-steps) of simulation. Chosen models are the best tested models for 3 size of retained POD modes. The visualisation is divided into 3 lines, the first shows prediction in the full physical space, the middle line shows prediction in the subspace orthogonal to the retained subspace, and the last line shows prediction in the POD subspace. Blue curves show models' predictions and red curves show ground truth simulation.

5.3. Model analysis on the Burgers' equation test case

Analysis of model's output in each subspace. Figure 5 illustrates the model's predictive capabilities across various test set examples. Predictions and ground truth are decomposed into subspaces: the bottom shows the filtered solution in the POD subspace, the middle filters to the AE subspace, and the top combines both to reconstruct the full space solution. We highlight three key findings. First, the qualitative results show near-perfect reconstruction in all three subspaces, indicating our model achieves high accuracy. Second, the trajectory norm within the unretained subspace is notably large, particularly for models with 5 and 10 POD modes, underscoring the relevance of our AE approach. Finally, the visualisation reveals that the small remaining error lies primarily in the subspace orthogonal to the POD, which is desirable, as this component corresponds to the physically unexplained part of the model.

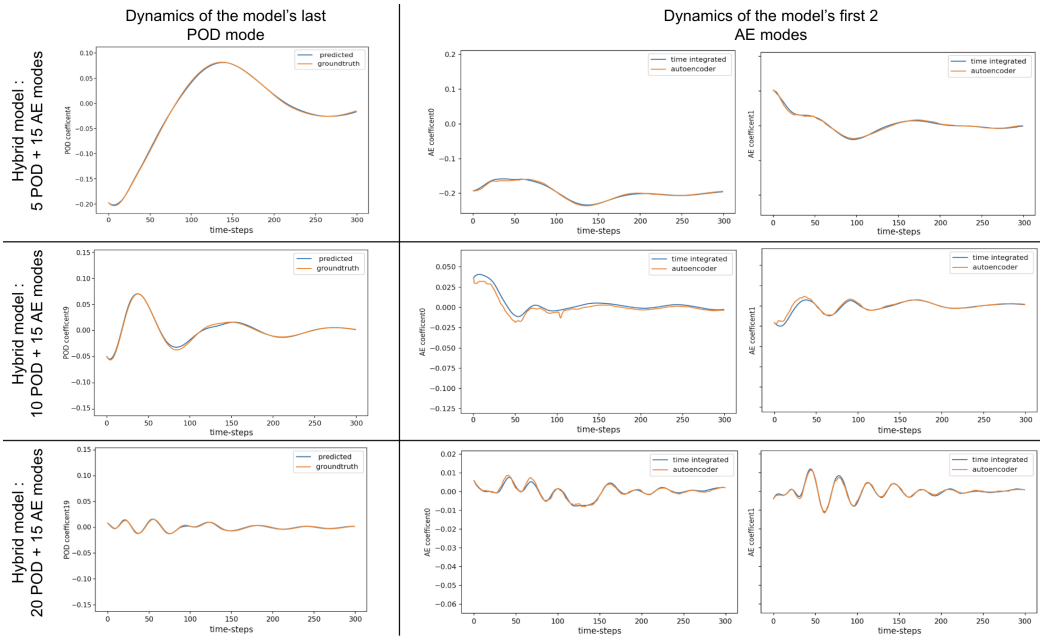


Figure 6: Plot of the time dynamics in the hybrid latent space for 3 examples of the test set. The left column depicts the time dynamics for each model's last POD mode. The blue curve represents our model's predicted dynamics, and the orange curve represents the ground truth. The two right columns show the dynamics for the first 2 AE modes. The blue curve shows the dynamics given by the time integrator neural network, and the orange curve depicts the encoded representation of the ground truth dynamics.

Latent space dynamics. Figure 6 illustrates the latent space dynamics for three test set examples using our best models with 20, 10, and 5 POD modes. The POD subspace (left column) shows near-perfect reconstruction of POD coefficients across all models, confirming that the main sources of error lie within the less interpretable AE subspace. Notably, the dynamics in the AE subspace (two right columns) are unusually smooth for an autoencoder. The joint training of the AE and time integrator appears to regularise the autoencoder mapping. In addition, the AE mode dynamics exhibit temporal patterns resembling those of the final retained POD modes, both in amplitude and frequency content. This POD-like behaviour, which emerges without explicit regularisation, is particularly noteworthy. Finally, the AE mode plots display two curves: one predicted by the time integrator, and one representing the encoded ground truth trajectory. Even in the least accurate examples, the smoothness and consistency between these neural representations remain acceptable.

Optimal latent space composition. We analyse the impact of latent space composition in Figure 7, beginning with the investigation of optimal ratios for latent space construction. Each curve corresponds to a model with a fixed number of POD modes: 0, 5, 10, and 20. Increasing the number of AE modes beyond 10–15 yields diminishing performance gains, while adding POD modes consistently improves accuracy. Adopting a complementary view in Figure 8, we fix the total latent space size to compare the reduction capabilities of different models. The comparison includes 18 variants of our hybrid models, fully data-driven models, classical POD ROMs, and the POD projection error, which serves as an upper bound for methods constrained to the POD subspace.

Classical ROMs perform poorly, likely due to the problem's strongly convective and nonlinear character, and because the POD basis is built from a high-variance dataset. When trained on a single example, the same ROM performs better but fails to generalise effectively across broader datasets. The results show that hybrid models achieve the best performance for moderately large latent spaces (15 to 50 modes). In contrast, fully data-driven methods are more effective when the latent space is small (10 modes or fewer), possibly due to the higher information density provided by autoencoders.

Computational cost. The proposed method clearly outperforms traditional POD-Galerkin ROMs in accuracy at equivalent computational cost. Specifically, it achieves over two orders of magnitude lower MSE ($\text{MSE} = 8.4 \times 10^{-6}$)

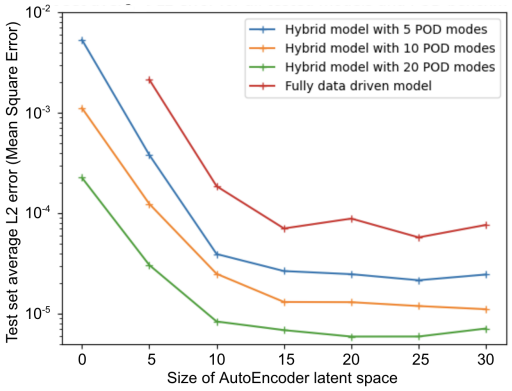


Figure 7: Graph showing models' performance based on the number of autoencoder modes.

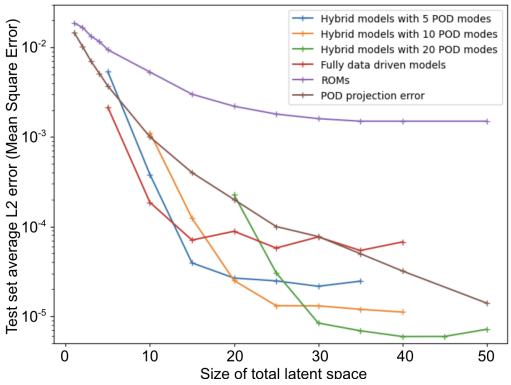


Figure 8: Graph showing all models' performance based on the total size of their reduced space.

compared to a 35-mode POD-Galerkin ROM (MSE $= 1.5 \times 10^{-3}$). The computational cost of our model is 1.23×10^8 floating point operations (FLOPs) per trajectory integration (300 time steps), closely matching the 1.25×10^8 FLOPs of the 35-mode ROM. Beyond 35 modes, ROM performance on the test set stagnates, showing that increasing the cost of a traditional POD-Galerkin ROM cannot match our model's accuracy. This confirms that our hybrid method offers significantly higher accuracy with no additional computational overhead. Although no speed-up over the FOM is observed in this numerically simple test case, the next case (cylinder flow) is expected to highlight this advantage. Training time and the hyperparameter tuning used to obtain the reported performance are detailed in Appendix A, supporting the method's relevance for multi-query scenarios, where its high accuracy and efficiency justify the training cost.

6. Single trajectory cylinder flow test case and results

6.1. Test case description

For our initial 2D experiment, we study the Navier-Stokes equations (equation 20) by simulating cylinder flow at Reynolds number 100.

$$\begin{cases} \dfrac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} - \nu \Delta \mathbf{u} + \nabla p = \mathbf{f} \\ \\ \nabla \cdot \mathbf{u} = 0 \end{cases} \tag{20}$$

This test case is a classical fluid dynamics benchmark [56], known for its vortex shedding and moderate complexity. At Reynolds number $Re = 48$, the flow undergoes a Hopf bifurcation, transitioning from a steady base flow to a periodic limit cycle—the Von Kármán vortex street—characterised by alternating vortex shedding. The base flow instability and the well-documented periodic behaviour of the limit cycle make this case well-suited for evaluating reduced-order models (ROMs), which must capture both the instability growth rate and vortex shedding frequency.

To represent the flow's essential dynamics, we restrict the latent space to three POD modes: one shift mode for the mean flow and two oscillatory modes capturing the shedding dynamics [57, 58]. These modes are standard in ROM studies for their balance between compactness and physical interpretability. Simulations are performed using FreeFEM++ [59], with a mixed-element method employing P1-bubble velocity elements and first-order pressure elements. This discretisation yields 27,000 degrees of freedom (see Figure 9). A first-order time-stepping scheme is used, with $\Delta t = 0.04$ s and 4000 time steps, covering the transition from base flow to several limit cycle periods.

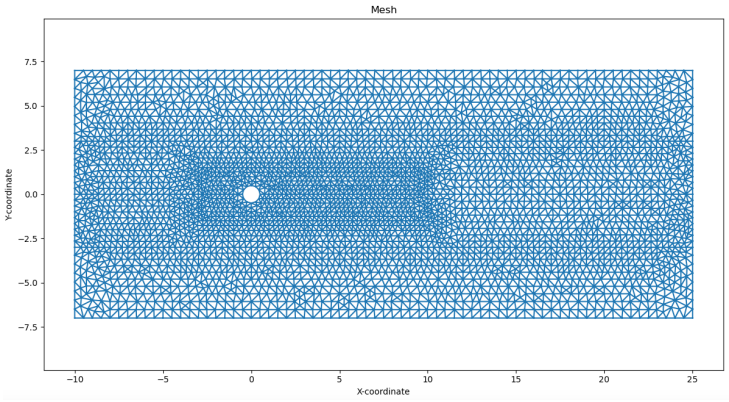


Figure 9: Mesh for cylinder flow test case

6.2. Single trajectory cylinder flow results

Qualitative results. Our analysis of the model's qualitative performance focuses on two aspects: prediction accuracy within the latent space of retained POD modes, and reconstruction of dynamics from the discarded information encoded by the autoencoder modes.

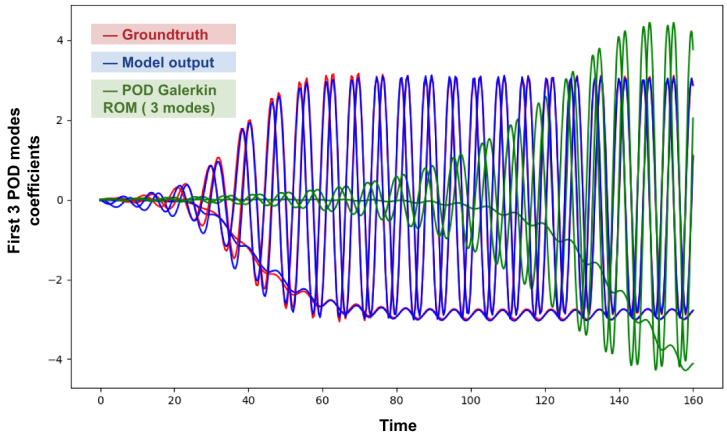


Figure 10: time evolution of the 3 retained POD modes coefficients when comparing our hybrid strategy to a 3 mode POD Galerkin ROM and the ground truth data.

Figure 10 shows the latent space dynamics of the three retained POD modes, comparing our hybrid model (3 POD + 27 AE modes) with the full-order model (ground truth) and a 3-mode POD Galerkin ROM. This visualisation highlights the model's ability to capture key dynamical features, including limit cycle amplitude, frequency, and instability growth rate. The results demonstrate that the hybrid model clearly outperforms the traditional POD Galerkin ROM. Although the latter correctly predicts the limit cycle frequency, it fails to capture the transient dynamics leading to the limit cycle and inaccurately represents both amplitude and growth rate. In contrast, the hybrid model accurately reproduces the large-scale dynamics: the inclusion of AE information and closure modelling allows it to precisely match the instability growth rate from the base flow, as well as the amplitude, frequency, and phase of the limit cycle.

To evaluate the model's performance beyond large-scale dynamics, we examined information outside the POD subspace. Figure 11 highlights the significant content in this subspace, supporting our use of an autoencoder. The figure compares the ground truth with the model output at a point on the limit cycle, filtered to retain only the unrepresented POD subspace. The shape and magnitude of small-scale structures are accurately reconstructed. As this snapshot occurs after the solution has reached the limit cycle, the results show that the model not only predicts the correct AE dynamics but also reconstructs them with precision. This dual capability in prediction and reconstruction demonstrates the effectiveness of our hybrid approach in

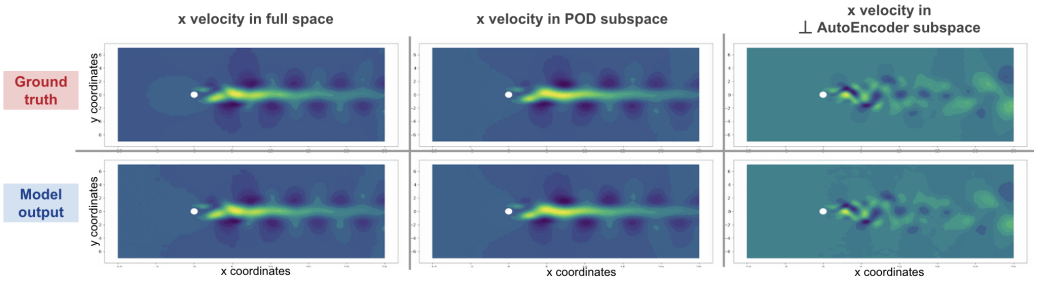


Figure 11: Visualisation of the model's accuracy decomposed in POD and autoencoder subspace for a snapshot on the limit cycle.

capturing the full complexity of the cylinder flow at $\mathrm{Re} = 100$.

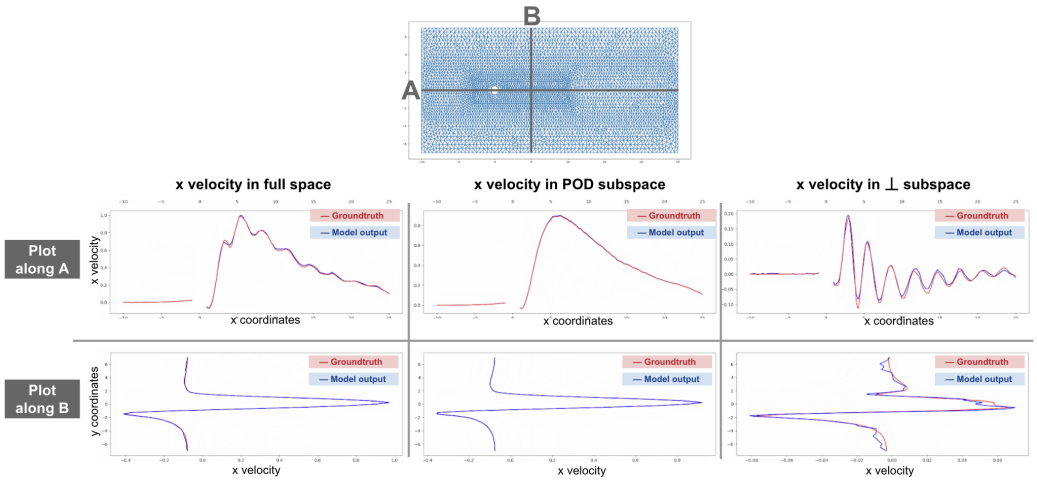


Figure 12: Visualisation of the model's accuracy along two cuts of the domain and decomposed in full, POD and Autoencoder subspace for a snapshot on the limit cycle.

Quantitative results. To assess the quantitative performance of our model, we focus on two aspects: the prediction of horizontal velocity along selected domain cross-sections and the overall MSE comparison across models for the full trajectory. Figure 12 shows the two cross-sections used for velocity analysis: a horizontal cross-section across the domain width at mid-height, and a vertical cross-section just downstream of the cylinder. The plots below display horizontal velocity predictions along these sections after 4000 simulation steps, when the flow has reached its limit cycle. The results support our qualitative observations. Within the POD subspace, our model's predictions

Model	test-set MSE
Hybrid model 3 POD + 27 AE modes	$\mathbf{1.5 \times 10^{-5}}$
POD-Galerkin ROM 3 modes	2.64×10^{-2}
POD-Galerkin ROM 30 modes	1.68×10^{-4}
3 modes POD projection error	1.51×10^{-3}

Table 3: Mean square error (MSE) on the complete time integration of the cylinder flow at $Re = 100$ for hybrid models and comparative models with 30 modes total latent size.

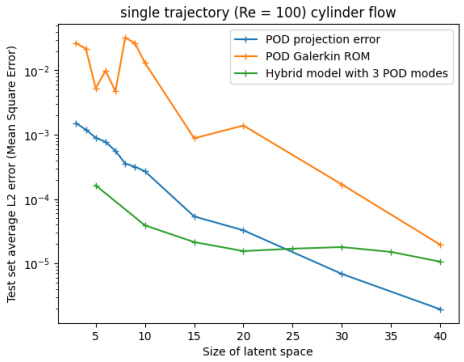


Figure 13: Comparison of the hybrid model performance and ROM performance with respect to the size of the retained subspace

are nearly indistinguishable from the ground truth. In the orthogonal complement, we observe excellent agreement along the horizontal cut and slightly less accurate but still strong predictions for the vertical cut. Overall, flow structure values in the orthogonal subspace are well captured, with only minor discrepancies near the boundaries. These results confirm that our model accurately reproduces flow structures and achieves quantitative performance close to the ground truth. Table 3 reports the MSE averaged over all time steps for various models, offering a broader view of performance. Our hybrid model significantly outperforms both the 30-mode POD-Galerkin ROM and the projection error of a 3-mode POD. This means it also surpasses models that correct only within the POD subspace, such as the LSTM ROMs and CD-ROM discussed in Section 2.1. To better understand performance trends and identify the optimal number of AE modes, we also plot the MSE as a function of latent space size in Figure 13. Our model consistently outperforms all similarly sized ROMs. For latent spaces smaller than 20, it even outperforms the POD projection error, thereby exceeding all models constrained to the POD subspace. This is particularly notable since the cylinder flow trajectory is well suited to POD—unlike the Burgers case—yet our model still delivers superior results.

Assessment of model stability and robustness to unseen initial conditions. The previous sections have demonstrated the accuracy of our method. To further evaluate its performance, we assess the model's stability and robust-

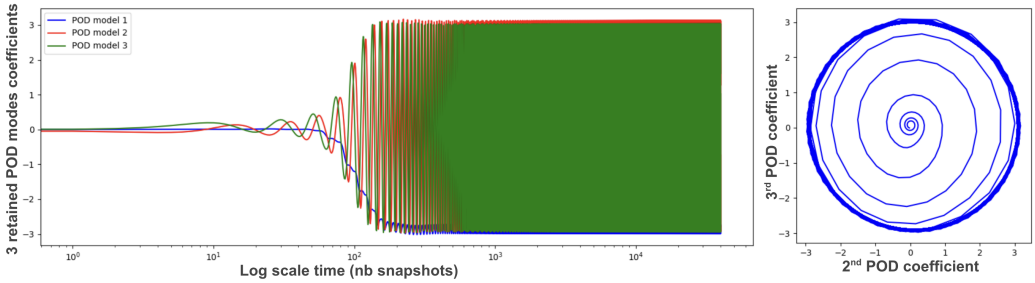


Figure 14: Left : Dynamics of the 3 retained POD coefficients for an inference trajectory 100 times longer than the training trajectory.
Right : Oscillatory POD modes phase plot of the same trajectory

ness during inference. While the linear component of our model ensures a physical foundation, the deep learning part does not guarantee reliable responses to unseen data. We therefore test its behaviour in unfamiliar regimes through three test-time experiments on the cylinder flow case. First, we examine whether the model maintains accuracy over horizons far beyond the training trajectory. As shown in Figure 14, when testing on a trajectory 100 times the training duration, the hybrid solution remains on the limit cycle without diverging. This indicates the model can generate consistent long-horizon predictions, motivating a deeper analysis of its local stability.

To further assess stability, we perturb the system at a point on the limit cycle, considering two regimes: small perturbations to evaluate the linear response and trajectory-tracking ability, and larger perturbations to test recovery under nonlinear effects. The procedure is identical for both, differing only in perturbation magnitude. Starting from the base flow, the system is integrated until it reaches the limit cycle, continuing for three-quarters of the training trajectory. At this point, Gaussian noise is added to the latent representation: $\mathcal{N}(\mu = 0, \sigma = 0.1)$ for small perturbations and $\mathcal{N}(\mu = 0, \sigma = 2)$ for large ones. The standard deviation is uniform across all latent space dimensions. For each regime, we generate 128 perturbed trajectories. Figures 15 and 17 show the system's response to small and large perturbations, respectively. In both cases, all trajectories return to the limit cycle within a reasonable time frame. This is evident in the time series envelopes of the three retained POD modes and the phase plots, where all red-coloured trajectories eventually align with the limit cycle, indicating stable long-term behaviour.

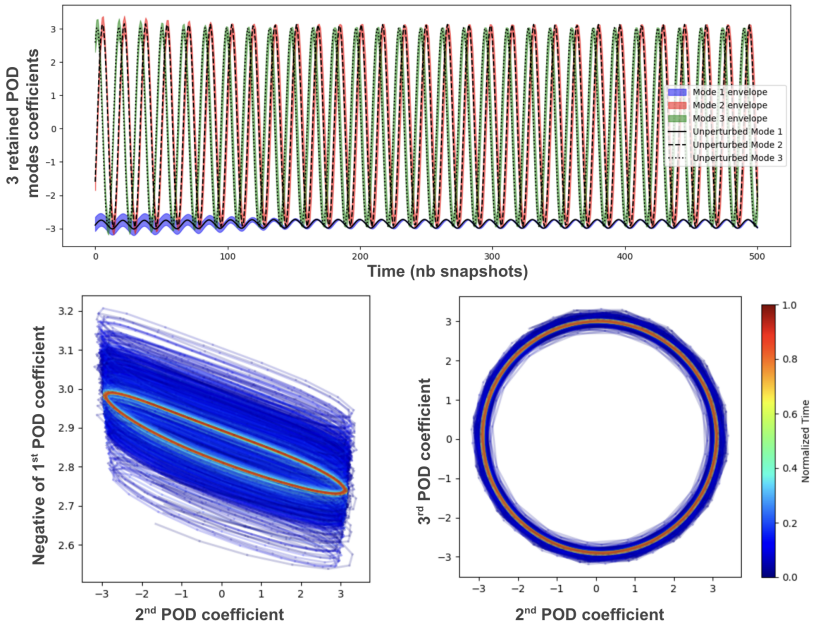


Figure 15: Top: Time evolution of the first three POD mode coefficients for 128 small amplitude perturbations.
Bottom: Phase plots of the same trajectories. Left: Mode 1 (Y) vs. Mode 2 (X). Right: Mode 2 (X) vs. Mode 3 (Y). Jet colormap indicates transient evolution (blue → green) and approach to the limit cycle (green → red).

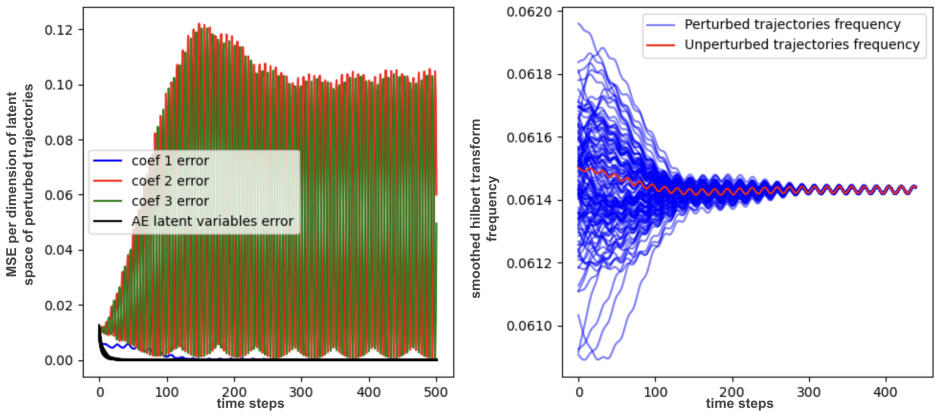


Figure 16: Left: Small amplitude perturbation decay per latent dimension.
Right : Hilbert transform post-processed frequency for small amplitude perturbations

To quantify the system's response, we compared the perturbed and unperturbed dynamics of each POD mode using the average L^2 norm of their differences. The results in Figure 16 (left) and Figure 18 (left) show that most modes exhibit exponential decay. However, the second and third POD modes—representing the primary oscillatory components—initially diverge and then plateau at a constant error. Since all trajectories eventually return to the limit cycle, this behaviour suggests a transient frequency mismatch rather than amplitude divergence.

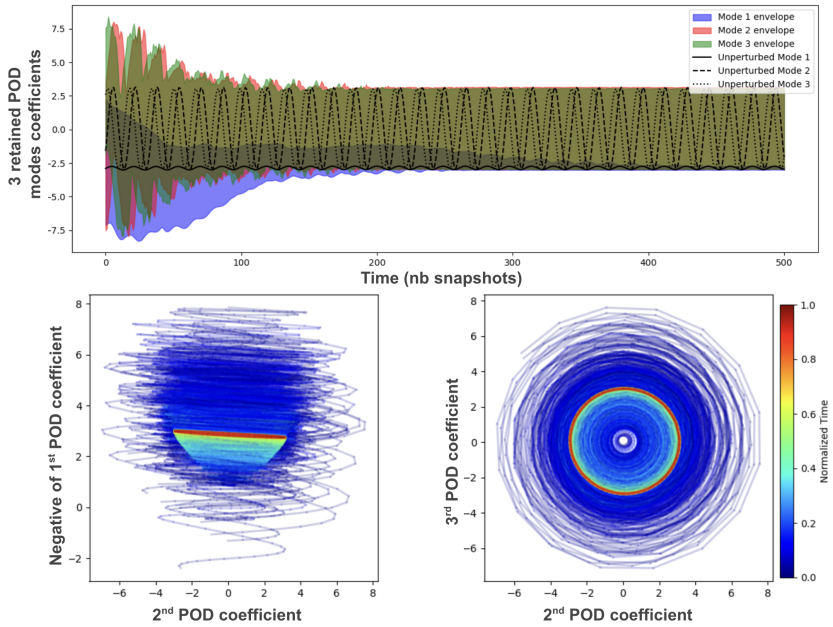


Figure 17: Top: Time evolution of the first three POD mode coefficients for 128 large amplitude perturbations.
Bottom: Phase plots of the same trajectories. Left: Mode 1 (Y) vs. Mode 2 (X). Right: Mode 2 (X) vs. Mode 3 (Y). Jet colormap indicates transient evolution (blue → green) and approach to the limit cycle (green → red).

To verify this hypothesis, we applied a Hilbert transform to the second and third POD modes to extract their instantaneous frequencies. Figure 16 (right) and Figure 18 (right) display the frequency evolution for small and large perturbations, respectively. In both cases, the perturbed trajectories (blue) exhibit an initial transient frequency shift relative to the unperturbed reference (red). For small perturbations, the maximum frequency difference reaches 0.8%; for large perturbations, it increases to 16%. However, this shift

is transient, and as the dynamics return to the limit cycle, the frequencies stabilise at their correct values. This behaviour is expected, as the system no longer lies exactly on the limit cycle immediately after a perturbation. For the cylinder flow the vortex shedding frequency depends slightly on the flow state, with frequencies near the base flow differing from those on the limit cycle. Thus, perturbations temporarily modify the system's oscillatory dynamics until it re-stabilises at the limit cycle. This indicates that the model captures physically plausible dynamics and has learnt that frequency varies with the system's distance from the limit cycle.

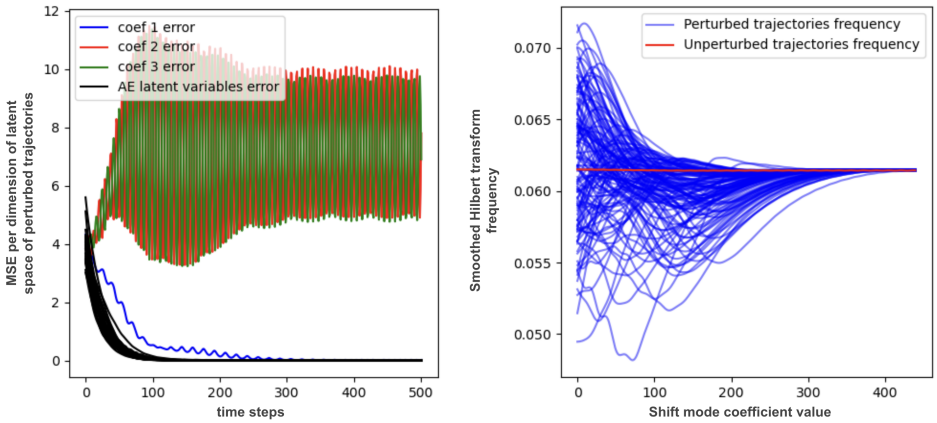


Figure 18: Left: Large amplitude perturbation decay per latent dimension.
Right : Correlation plot of Hilbert transform instantaneous frequency with corresponding shift mode value

Finally, we evaluate the model's robustness to perturbed initial conditions, focusing on the transient regime where perturbations may grow at different rates compared to those applied directly on the limit cycle. This experiment also assesses the encoder's sensitivity to noise, with noise added after POD encoding, thereby impacting both dynamics prediction and the encoding–decoding process. Two noise magnitudes are considered: $\mathcal{N}(\mu = 0, \sigma = 0.01)$ and $\mathcal{N}(\mu = 0, \sigma = 5)$ and applied on the large 2 tier POD coefficients and not the full physical space to prevent unwanted filtering by the POD. For small perturbations (Figure 19), the predicted flow closely follows the reference trajectory, indicating that minor disruptions do not impair accuracy. For larger deviations, which exceed the training regime by over an order of magnitude (Figure 20), the hybrid model still produces physically consistent transients and converges to the correct limit cycle in

reasonble time. This is especially notable given the training on a unique trajectory which would normally promote overfitting.

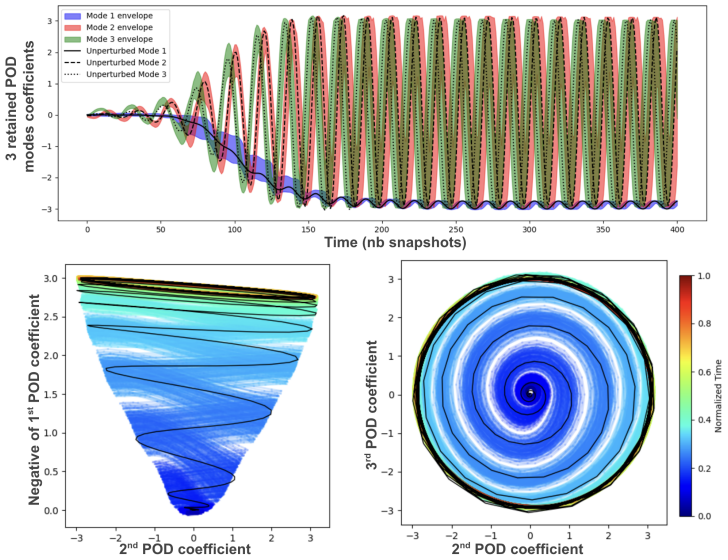


Figure 19: Top: Time evolution of the first three POD mode coefficients for 128 trajectories under unseen initial conditions, color-coded by mode (Mode 1: blue, Mode 2: red, Mode 3: black).
Bottom: Phase plots of the same trajectories. Left: Mode 1 (Y) vs. Mode 2 (X). Right: Mode 2 (X) vs. Mode 3 (Y). Jet colormap indicates transient evolution (blue → green) and approach to the limit cycle (green → red).

Computational cost. To evaluate the computational efficiency of our hybrid POD/deep learning model, we compare it with both the FOM and a traditional POD-Galerkin ROM, focusing on two objectives: achieving significant speed-up over the FOM and achieving lower computational cost than an equivalent-performance ROM. Both are assessed using two metrics: computational time and floating-point operations (FLOPs), the latter providing a hardware-independent measure of effort. For the first objective, our model reduced compute time from 163s (FOM on Xeon(R) Gold 5118 CPU) to 11s (hybrid model on Tesla V100S GPU), and operations from 1.5×10^{12} to 2.3×10^{9}, achieving a speed-up of 15 to 500 times. This supports the model's scalability for more complex cases, as hypothesised in the Burgers equation results. For the second objective, we compare with the smallest ROM achieving comparable (though slightly higher) error. The cylinder flow test case,

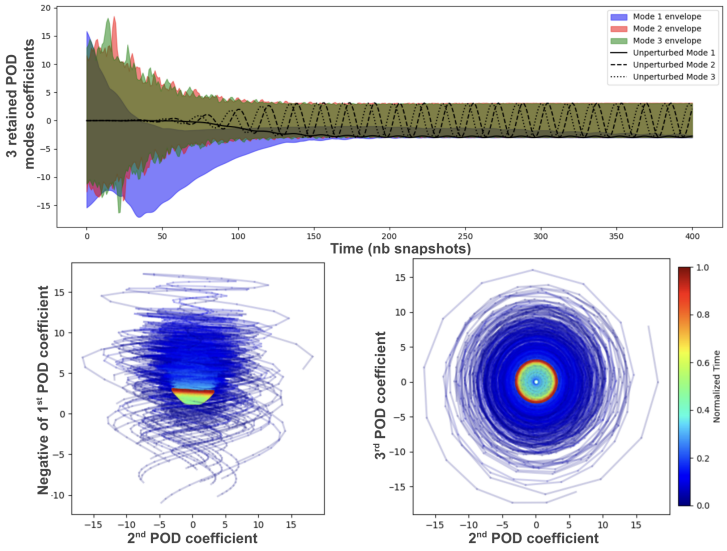


Figure 20: Top: Time evolution of the first three POD mode coefficients for 128 trajectories under unseen initial conditions, color-coded by mode (Mode 1: blue, Mode 2: red, Mode 3: black).
Bottom: Phase plots of the same trajectories. Left: Mode 1 (Y) vs. Mode 2 (X). Right: Mode 2 (X) vs. Mode 3 (Y). Jet colormap indicates transient evolution (blue → green) and approach to the limit cycle (green → red).

characterised by smooth periodic dynamics, is well suited to POD-Galerkin ROMs; thus, any speed-up in this setting is noteworthy. Our hybrid model yields a modest advantage, with fewer operations (2.3×10^9 vs 3.2×10^9), corresponding to a 1.5x speed-up. Additionally, earlier results on the Burgers equation showed a widening performance gap in scenarios less favourable to standard POD-Galerkin ROMs. These findings align with theoretical expectations. Our model combines two computational components: the ROM and a neural network. The ROM scales cubically ($\mathcal{O}(n^3)$) with the number of POD modes, while the neural network scales linearly ($\mathcal{O}(n)$) with the number of autoencoder modes. By reducing the number of required POD modes via autoencoder modes, our hybrid model shifts the computational burden from cubic to linear scaling. This becomes increasingly beneficial as problem complexity and mode count rise, where traditional ROMs are hindered by cubic cost. Full details on training time and hyperparameter tuning are given in Appendix A. Moreover, as shown in the previous subsection, the model's performance outside the training regime can be weighed against training cost, showing particular value in multi-query scenarios.

7. Fluidic pinball in the quasi-periodic regime

7.1. Test case description

We now consider a second 2D test case, the fluidic pinball, chosen to highlight a different aspect of the proposed model. While sharing several features with the cylinder flow test case, it exhibits substantially higher sensitivity to parameters and modelling errors. The configuration consists of three identical cylinders of equal diameter, positioned at the vertices of an equilateral triangle in close proximity, as shown in Figure 21. This arrangement generates strong wake interactions and various flow regimes as the Reynolds number changes.

For $Re \approx 15$ to $Re \approx 120$, the fluidic pinball passes through four distinct dynamical regimes, each marked by qualitative changes in wake structure and stability [60]. This study focuses on the regime for $Re \in [100, 115]$, where the flow is quasiperiodic, featuring two incommensurate shedding frequencies that form a torus-like attractor in phase space. For $Re > 115$, the system transitions to chaotic dynamics.

Capturing the dynamics in this quasiperiodic regime is challenging partly due to the small parameter range. Even small modelling errors—particularly from intrusive, truncated Galerkin models—can destabilise the solution and

prematurely drive the dynamics into chaos. Although the modal energy spectrum decays relatively fast, allowing a low-dimensional POD space to represent the state field, this alone does not ensure accurate temporal dynamics. As shown later, projection based reduced models fail to remain on the correct quasiperiodic attractor and predict the wrong regime.

Simulations are performed with FreeFem++ [59]. The mesh uses P1-bubble velocity elements and first-order pressure elements, with approximately 200,000 degrees of freedom due to the complex geometry. Time integration employs a second-order multi-step scheme with $\Delta t = 0.001$, viscosity $\nu = 10$, and inlet velocity $u = 11$. We simulate 35,000 time steps on the attractor to capture multiple quasiperiodic periods. Snapshots are saved every 50 time steps, yielding 700 training snapshots. The test dataset spans 25,000 time steps at the same saving rate, producing 500 test snapshots.

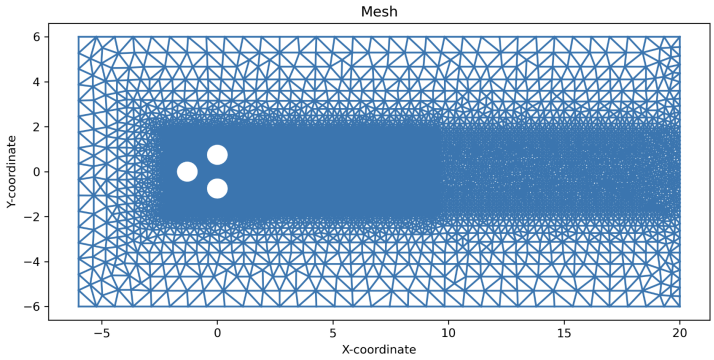


Figure 21: Mesh for the fluidic pinball test case, featuring three cylinders in an equilateral triangle configuration.

7.2. Qualitative Results for the Fluidic Pinball

The objective of this test case is to demonstrate our model's capacity to address the limitations of projection based reduced-order models (ROMs) affected by truncation effects. In models relying solely on the physics of large yet truncated Galerkin subspaces, errors can destabilise the dynamics, producing inaccurate or even chaotic predictions.

For this case, our hybrid model uses a two-tier POD decomposition with 100 total POD modes. The first tier comprises the 10 most energetic modes, integrated linearly. The remaining 90 modes are compressed into a 10-dimensional latent space via an autoencoder. The resulting 20-dimensional

latent space thus combines large-scale dynamics (first 10 dimensions) with nonlinear corrections from the compressed subspace (second 10 dimensions).

We compare our hybrid method to two baselines:

- A 100-mode POD-Galerkin ROM retaining 99.995% of the total energy. Although this truncation level exceeds that typically used in ROM studies, it remains insufficient to capture the quasiperiodic dynamics accurately.
- A fully data-driven model consisting of a 100-mode POD projection, followed by an encoder - Neural ODE time integrator - decoder architecture with a 20 variable latent space.

Figure 22 shows the time evolution of the first five POD mode coefficients for the test trajectory, illustrating each model's ability to reproduce the dynamics. Additionally, Figure 23 compares the predicted lift (C_L) and drag (C_D) coefficients, reconstructed from the 100-dimensional POD basis, providing a global measure of the flow dynamics and pressure reconstruction.

The results show that the POD-Galerkin ROM fails to capture the dynamics despite the high energy content of its basis. Its predictions diverge rapidly from the ground truth, losing the quasiperiodic structure and instead exhibiting behaviour akin to a transition to chaos—likely due to incorrect modelling of nonlinear interactions with discarded modes. In contrast, the fully data-driven model yields a coherent quasiperiodic signal, albeit with slight amplitude discrepancies. Our hybrid model avoids these issues and qualitatively reproduces the true dynamics, both in POD mode evolution and in global force coefficients. These results highlight the need for nonlinear modelling and demonstrate the effectiveness of combining a physics-based ROM structure with a data-driven autoencoder correction, particularly for complex flows where truncation effects are significant.

7.3. Quantitative Results for the Fluidic Pinball

To complement the previous section's observations, we assess the models quantitatively using the mean relative error on the test trajectory:

$$\varepsilon = \frac{1}{N} \sum_{k=1}^{N} \frac{\|\mathbf{a}_{\text{true}}^{(k)} - \mathbf{a}_{\text{pred}}^{(k)}\|_2}{\|\mathbf{a}_{\text{true}}^{(k)}\|_2}$$

where N is the number of snapshots, and $\mathbf{a}^{(k)}$ is the reduced-order state at time step k.

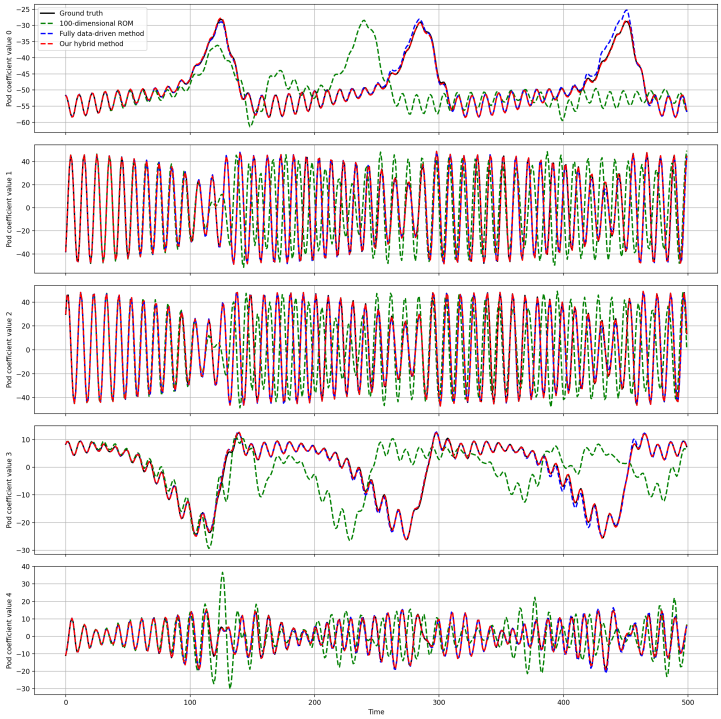


Figure 22: Time evolution of the 5 leading POD modes for four models: the full-order model (black, ground truth), our hybrid model (red dashed line), the 100-dimensional POD-Galerkin ROM, and a fully data-driven model with the same latent space size as our hybrid model.

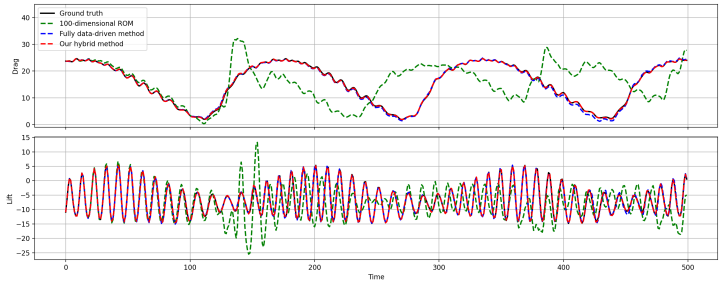


Figure 23: Time evolution of the lift (C_L) and drag (C_D) coefficients for four models: the full-order model (black, ground truth), our hybrid model (red dashed line), the 100-dimensional POD-Galerkin ROM, and the fully data-driven model with the same latent space size as our hybrid model.

Table 4 lists the mean relative error for the three models considered. The subspace size corresponds to the dimension of the latent space or Galerkin basis. The results corroborate the qualitative trends: our hybrid model is about 3.4 times more accurate than the fully data-driven NODE model and over 45 times more accurate than the POD-Galerkin ROM. This highlights the benefit of combining a physics-based subspace with a nonlinear data-driven correction to preserve both accuracy and stability.

Model	Subspace Size	Mean Relative Error
Hybrid (10 POD + 10 AE)	20	$\mathbf{1.69 \times 10^{-2}}$
Data-driven (20 AE)	20	5.79×10^{-2}
POD-Galerkin ROM	100	7.67×10^{-1}

Table 4: Quantitative comparison of the models on the quasi-periodic fluidic pinball test set.

8. Parametric cylinder flow test case

8.1. Parametric Problem Description

The first part of our study addressed fixed parameters, limiting the model's generalisability. Extending to parametric problems introduces additional challenges but enhances real-world applicability, as variations in parameters—such as the Reynolds number—can cause significant, nonlinear changes in flow dynamics. Parametric modelling is particularly relevant for applications such as shape optimisation [61], data assimilation [62], and surrogate modelling for engineering design exploration [63]. Capturing system behaviour across a range of parameters is essential for predictive reliability.

We consider a family of nonlinear PDEs with parameter-dependent dynamics, expressed in semi-discrete weak form as:

$$M d_t \mathbf{w} = \mathcal{L}(\nu)\mathbf{w} + \mathcal{B}(\mathbf{w}, \mathbf{w}), \tag{21}$$

where M is the mass matrix and $\mathbf{w}$ the high-dimensional state vector, satisfying homogeneous boundary conditions. The dynamics are governed by a linear operator $\mathcal{L}(\nu)$ and a bilinear nonlinear operator $\mathcal{B}$.

Although the parameter dependence lies in the linear term $\mathcal{L}(\nu)$, its variation with respect to the parameter is nonlinear. This relation affects most the learnt dynamics correction and its effects are detailed in Appendix B, where we present numerical evidence that the reduced model exhibits a nontrivial, nonlinear dependence on ν.

8.2. Reduced-order modeling for the parametric problem

We adopt the perturbative formulation introduced above to handle parameters (e.g., Reynolds). For each parameter value ν, the base flow $\mathbf{w}_0(\nu)$ solves the steady Navier–Stokes problem and varies *nonlinearly* with ν. The perturbation $\mathbf{w}'$ then satisfies

$$\begin{aligned} M d_t \mathbf{w}' &= \mathcal{L}'(\nu)\mathbf{w}' + \mathcal{B}(\mathbf{w}', \mathbf{w}') \\ \text{with} \quad \mathcal{L}'(\nu)\mathbf{w}' &= \mathcal{L}(\nu)\mathbf{w}' + \mathcal{B}(\mathbf{w}_0(\nu), \mathbf{w}') + \mathcal{B}(\mathbf{w}', \mathbf{w}_0(\nu)) \end{aligned} \tag{22}$$

Even if $\mathcal{L}(\nu)$ is affine in ν, the dependence of $\mathcal{L}'(\nu)$ is generally *non-affine* because $\mathbf{w}_0(\nu)$ depends nonlinearly on ν.

For training, we assemble a parametric dataset of perturbation trajectories $\{\mathbf{w}'(t, \nu_m)\}_m$ at multiple ν_m, and a test set at previously unseen parameters. A single global POD basis Φ_r is computed from the concatenated training snapshots to capture dominant structures across all ν_m, trading some reconstruction accuracy for cross-parameter generality; consequently, more modes may be required to satisfy a given energy threshold.

During training and testing, the parameter-dependent operator is projected onto the reduced subspace:

$$L'(\nu_m) \;=\; \Phi_r^\top \mathcal{L}'(\nu_m)\, \Phi_r \tag{23}$$

with Φ_r the truncated POD basis of rank r. The reduced perturbation dynamics read

$$d_t a_i \;=\; \mathrm{ROM}(a_i, \nu_m) \;=\; \sum_{j=0}^{r-1} L'_{ij}(\nu_m)\, a_j \;+\; \sum_{j,k=0}^{r-1} N_{ijk}\, a_j a_k \tag{24}$$

where a_i are the POD coefficients and $N_{ijk} = \boldsymbol{\varphi}_i^\top \mathcal{B}(\boldsymbol{\varphi}_j, \boldsymbol{\varphi}_k)$ are the (parameter-independent) quadratic interaction coefficients. Although (24) formally encodes parameter effects via $L'(\nu)$, the non-affine dependence induced by $\mathbf{w}_0(\nu)$ implies that the learned correction in our hybrid model must also capture nonlinearity in ν, the importance of this consideration on our test case is investigated in Appendix B.

Alternative parametric-ROM strategies include local bases and interpolation of reduced operators [64, 65]. We favor a single global basis for simplicity, preserved orthogonality, and seamless integration with our hybrid approach, wherein deep learning corrects the limitations of the projected model under this non-affine parametric dependence.

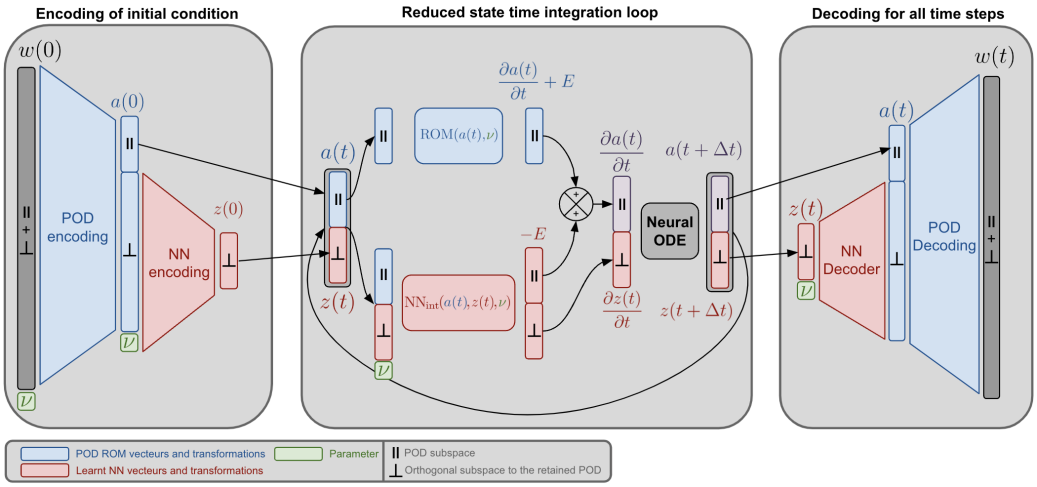


Figure 24: Model architecture diagram for the parametric variant of the model, ν represent the parameter and is used as input in all neural networks of the model and the reduced order model.

8.3. Model description for the parametric variant of the model

Encoder. The parametric extension of our model builds on the previously described architecture, with adjustments to incorporate parameter dependence. We denote the parameter as ν, scaled to match the range of the second-tier POD coefficients for learning stability. The encoder now takes ν as an additional input, following the conditional autoencoder approach of [66]. This enables the nonlinear encoding–decoding to vary across parameter values ν, improving the model's capacity to capture parameter-dependent states. The encoding process is given by:

$$\begin{aligned} \mathbf{a}_{\parallel}(t) &= \Phi_{\parallel}^{\top} M \mathbf{w}'(t), \\ \mathbf{z}(t) &= \mathrm{Encoder}(\mathbf{a}_{\perp}(t), \nu, \theta_E), \quad \mathbf{a}_{\perp}(t) = \Phi_{\perp}^{\top} M \mathbf{w}'(t), \end{aligned} \tag{25}$$

The full latent space is formed by concatenating the first-tier POD coefficients $\mathbf{a}_{\parallel}$, the encoder output $\mathbf{z}$, and the parameter ν. This design preserves interpretability and makes efficient use of the limited latent space, directing the encoder to represent only the unretained information rather than redundantly encoding the parameter.

Time Integration. Time integration in the hybrid model follows the non-parametric case, with one key change: the reduced linear operator becomes

parameter-dependent, obtained by projecting the full operator at each parameter value. This embeds parametric effects within the POD framework. The neural network takes the full latent state, including the parameter, as input and outputs the time derivatives of both POD and AE modes and parameter value is kept. The complete time integration system is:

$$\begin{aligned} d_t\hat{\mathbf{a}}_\parallel(t) &= \mathrm{ROM}(\hat{\mathbf{a}}_\parallel(t),\nu) + \mathrm{NN}_\parallel(\hat{\mathbf{a}}_\parallel(t),\hat{\mathbf{z}}(t),\nu,\theta_{NN}) \\ d_t\hat{\mathbf{z}}(t) &= \mathrm{NN}_\perp(\hat{\mathbf{a}}_\parallel(t),\hat{\mathbf{z}}(t),\nu,\theta_{NN}) \\ d_t\nu &= 0 \end{aligned} \tag{26}$$

Decoder. The decoding process retains the non-parametric structure for the POD part. The neural network decoder, however, now includes ν as an additional input, following standard practice in conditional autoencoders [67]:

$$\hat{\mathbf{w}}(t) = \Phi_\parallel\hat{\mathbf{a}}_\parallel(t) + \Phi_\perp\hat{\mathbf{a}}_\perp(t), \quad \hat{\mathbf{a}}_\perp(t) = \mathrm{Decoder}(\hat{\mathbf{z}}(t),\nu,\theta_D) \tag{27}$$

Normalisation of the loss for better model training. Model training follows the same procedure as in the non-parametric test case, except for the loss formulation. The overall structure is retained, but the reconstruction term is modified to address a scaling issue introduced by parametric test cases. Low Reynolds number flows generally have smaller coefficient values than high Reynolds number flows, leading to lower absolute norms of trajectories. Without scaling, loss minimisation yields disproportionately high relative errors for low Reynolds number flows and very small proportional errors for high Reynolds number flows. A similar effect occurs at the start of trajectories, near the base flow, where amplitudes are small and high relative errors correspond to small absolute errors. To mitigate this, we introduce a scaling factor in the loss formulation. Scaling is applied with respect to the parameter value and time, but not across the coefficient dimension, as we aim to preserve both relative and absolute accuracy for the most energetic modes.

The scaling factor is defined as:

$$\gamma(t) = \sqrt{\|\mathbf{a}_\parallel(t)\|^2 + \|\mathbf{a}_\perp(t)\|^2 + \epsilon} \tag{28}$$

where $\mathbf{a}$ denotes the POD coefficients of the current learning batch, arranged as a three-dimensional tensor: the first dimension is the example index, the second the degrees of freedom, and the third the time step, $\|\cdot\|$ denotes the L2 norm of the POD coefficients for each example over its timesteps, and ϵ is a small constant (e.g., 10^{-9}) for numerical stability.

With this scaling, the reconstruction term becomes:

$$L_{\text{recon}} = \frac{1}{N_t} \sum_{n=1}^{N_t} \frac{\left\|\hat{\mathbf{a}}_{\parallel}(t_n) - \mathbf{a}_{\parallel}(t_n)\right\|^2 + \left\|\hat{\mathbf{a}}_{\perp}(t_n) - \mathbf{a}_{\perp}(t_n)\right\|^2}{[\gamma(t_n)]^2} \tag{29}$$

This scaled formulation balances performance across Reynolds numbers and along the trajectory, ensuring consistent relative accuracy while maintaining focus on the most energetic modes.

8.4. Parametric cylinder flow test case

Our final 2D experiment extends the previous test case by introducing parametric variation to the cylinder flow problem. We examine flow dynamics for Reynolds numbers between 60 and 120, based on the Navier–Stokes equations (20) and using the same geometry as in the $Re = 100$ case. To better reflect real-world conditions, the Reynolds number is varied by adjusting the input velocity, mirroring practical situations where changes in flow usually result from varying speeds. The dataset comprises 31 simulations with Reynolds numbers defined as $Re(m) = 60 + 2m$, where m is an integer and $0 \leq m \leq 30$. For testing, we use a separate set with $Re(m) = 50 + 5m$ and $m \in \{0, 1, 3, 5, 7, 9, 11, 13, 15\}$. This choice enables assessment of the model's ability to interpolate between known parameters and extrapolate beyond the training range.

8.5. Parametric cylinder flow results

Qualitative results. Building on the previous single-Reynolds-number results, we demonstrate that, when trained on a dataset of cylinder flows at multiple Reynolds numbers, the model can predict trajectories both within and slightly outside the training range.

To assess accuracy across Reynolds numbers, we selected the second POD mode and applied a Hilbert transform to the predicted coefficients over time. This transform quantifies the instability growth rate towards the limit cycle and the oscillatory frequency. Three representative test examples are shown in Figure 25. In each example, the first row displays the dynamics of the second POD coefficient alongside the model prediction. The second row compares the frequency extracted via the Hilbert transform for ground truth and predicted data. While the Hilbert transform captures the growth rate and frequency of the periodic instability, artefacts occur at signal boundaries (Figure 25) and should be disregarded.

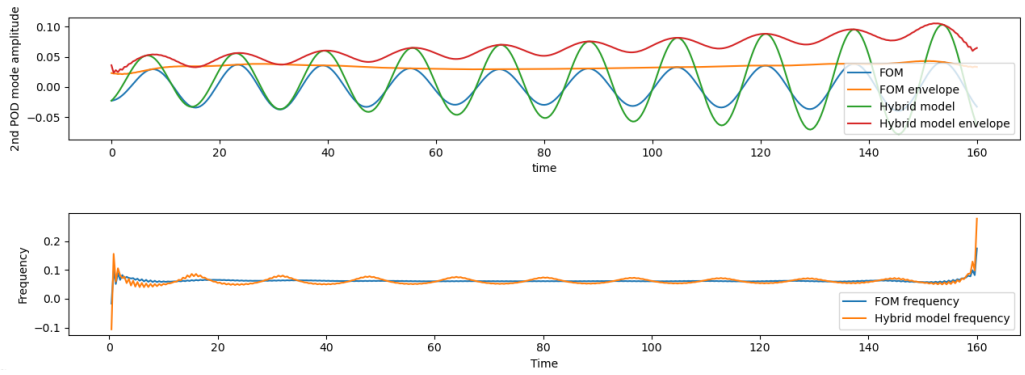
The 3 selected test examples are : the two most extreme Reynolds numbers, $Re = 50$ and $Re = 125$, both outside but near the training interval, and $Re = 105$, which closely matches the single-trajectory case discussed earlier. The model performs almost perfectly for $Re = 105$ and $Re = 125$ in terms of instability growth rate, phase, amplitude, and frequency. This performance is consistent across most Reynolds numbers, except for $Re = 50$ and $Re = 55$. In all cases, phase and frequency predictions are highly accurate, and the limit cycle envelope is generally well reconstructed, though minor growth rate errors remain.

Experiments suggest that inaccuracies at lower Reynolds numbers stem from the loss function formulation. For these cases, POD mode coefficients have smaller magnitudes, and since the MSE is computed on absolute error, such trajectories are underweighted during training. The same effect occurs for initial time steps, where POD mode norms are low. Furthermore, trajectories near the critical point show a nonlinearly slow instability growth rate, making accurate prediction of this growth particularly challenging, which explains the observed discrepancies.

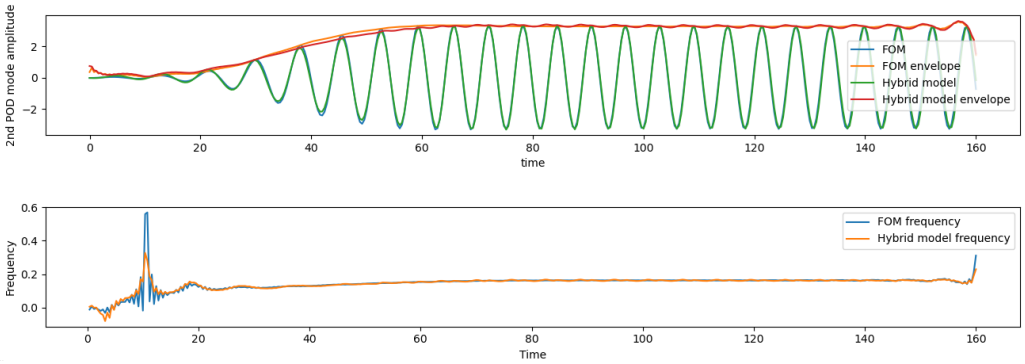
We also visualised the predicted trajectories in the physical space for both the POD and AE subspaces. Results closely match those for the 2nd POD coefficient. For Reynolds numbers in the range $Re = 60$ to $Re = 125$, the agreement is near perfect, as shown in Figure 26. This is evident in both the visualisations and the plots along cuts, for both the POD and autoencoder subspaces. These results are highly encouraging, indicating that the parametric variant of the model performs qualitatively as well as the non-parametric version, but here across a broader range of Reynolds numbers.

Quantitative results. The quantitative analysis of the model is consistent with the qualitative findings. Figure 27 shows the normalised MSE, obtained by dividing each example's predicted time step by the norm of the corresponding ground truth time step. Aside from the cases at $Re = 50$ and $Re = 55$, the loss on all unseen Reynolds numbers remains relatively stable, even for higher values such as $Re = 125$ outside the interpolation range. This stability demonstrates the model's robustness. In contrast, the noticeably higher relative MSE at $Re = 50$ and $Re = 55$ mirrors the qualitative differences observed in these flows. This trend is also evident in the aggregate performance in Table 5, where our hybrid model outperforms both 3-mode and 30-mode POD-Galerkin ROMs.

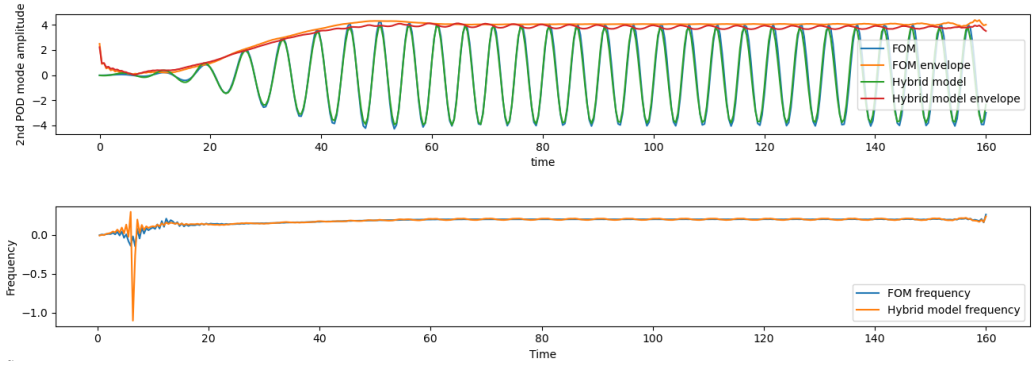
This discrepancy stems from three factors. First, these cases lie outside



(a) Second mode dynamics and Hilbert transform frequency and envelope post-treatment for extrapolate test set Reynolds = 50

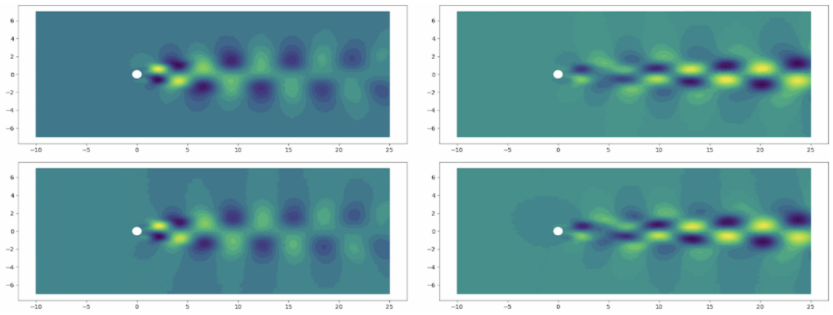


(b) Second mode dynamics and Hilbert transform frequency and envelope post-treatment for interpolated test set Reynolds = 105

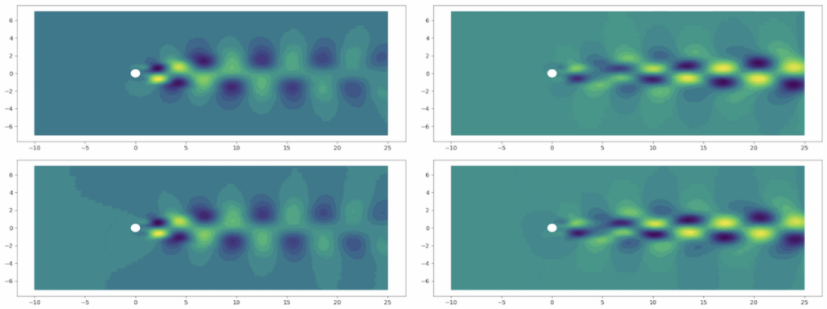


(c) Second mode dynamics and Hilbert transform frequency and envelope post-treatment for extrapolated test set Reynolds = 125

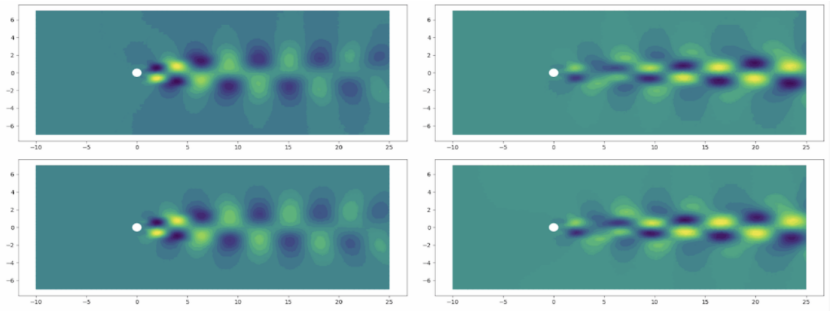
Figure 25: Second mode dynamics and Hilbert transform frequency + envelope post-treatment for various Reynolds numbers.



(a) Visualisation of POD and AE subspaces in physical space for test example at Reynolds 55 for final timestep



(b) Visualisation of POD and AE subspaces in physical space for test example at Reynolds 105 on the limit cycle



(c) Visualisation of POD and AE subspaces in physical space for test example at Reynolds 125 on the limit cycle

Figure 26: visualisation of POD and AE subspaces for ground truth and predicted snapshots, the top line represents ground truth, the bottom line represents the model prediction, the left column represents POD subspace and the right column represents the AE subspace.

Model	test-set MSE
Hybrid model 3 POD + 27 AE modes	$\mathbf{1.13 \times 10^{-4}}$
POD-Galerkin ROM 3 modes	1.42×10^{-2}
POD-Galerkin ROM 30 modes	6.53×10^{-3}
3 modes POD projection error	1.16×10^{-3}

Table 5: Mean square error (MSE) on the complete time integration of the cylinder flow averaged for all reynolds for hybrid models and comparative models with 30 modes total latent size.

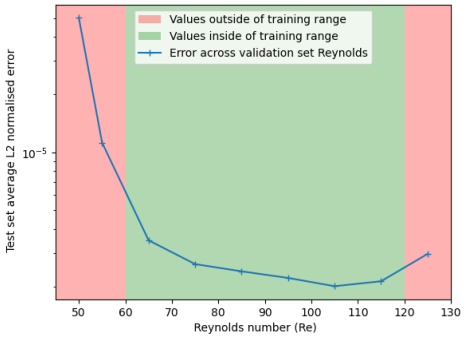


Figure 27: Plot of the relative MSE of the model for different Reynolds number test examples

the interpolation range and in a distinct dynamical regime, making them harder to predict. Unlike $Re = 125$, which extrapolates well due to similar flow behaviour to the training cases, $Re = 50$ and $Re = 55$ are both unseen and near the critical Reynolds number, where abrupt regime changes hinder extrapolation.

Second, the elevated relative loss reflects the impact of the chosen normalisation. We used pseudo-normalisation, dividing by the square root of the snapshot norm rather than the full norm, as this facilitated learning for high Reynolds number flows, where full normalisation proved less effective. While pseudo-normalisation partly mitigates uneven learning, it does not eliminate it, leading to slightly poorer results for lower Reynolds number trajectories.

Finally, the tuning of the loss normalisation reflects the emphasis in fluid dynamics on higher Reynolds number flows. Since current methods already perform well at $Re = 50$, our method's main contribution lies in delivering robust performance for higher Reynolds number trajectories.

9. Conclusion

Our hybrid approach, combining POD-Galerkin techniques with autoencoder-based neural networks, has demonstrated clear advantages across diverse non-chaotic test cases, providing a versatile framework for addressing multiple challenges in reduced-order modelling. In the Burgers equation, the model consistently outperformed traditional ROMs, particularly in high-frequency

regimes. For the cylinder flow problem at $Re = 100$, it accurately captured both large and fine scale structures, surpassing conventional POD-Galerkin ROMs and methods correcting only the POD subspace error. Extending the Reynolds number range from 60 to 120 further confirmed robustness, with accurate predictions across regimes and promising potential for real-world applications. Stability analysis reinforced this, showing reliable predictions over extended horizons and under substantial perturbations of initial conditions, with the model consistently recovering the correct limit cycle despite transient disturbances.

The study focused on non-chaotic systems, providing a stable setting to assess predictive capability. Current work is extending the approach to the fluidic pinball test case in the chaotic regime. This will test the model's forecasting ability on more intricate dynamics.

Our investigation into optimal latent space composition retained 18 variants for the Burgers equations, 8 for the cylinder flow test case, and another set for the parametric cylinder flow, totalling 27 variants, excluding those from hyperparameter tuning. This analysis clarified how latent space configurations affect performance. The chosen test cases posed varied challenges, from handling sharp fronts in the Burgers equation to capturing fine structures in higher Reynolds number flows, improving understanding of the method's capabilities.

The two-tier POD structure raises two questions: (1) At what resolution should the solution be predicted to exclude negligible noise or irrelevant scales? (2) What computational resolution is feasible for the first-tier POD-Galerkin ROM? Our approach offers flexibility to address this trade-off, with the deep learning component particularly effective for solutions with sharp fronts or fine details. By complementing the POD projection error, the method captures dynamics both within and beyond the chosen POD truncation. A capability valuable for complex flows where traditional ROMs struggle to represent the full range of dynamics.

An interesting future direction is replacing the standard POD-Galerkin ROM with hyper-reduction techniques such as ECSW, which offer substantial computational speed-ups and extend applicability to large-scale, general nonlinear systems. While hyper-reduction may introduce a small numerical error, its impact is limited if the reduced model preserves the dominant dynamics. In such cases, the ROM's role within the hybrid architecture remains unchanged, and the neural network's correction term can compensate for the added approximation. This work thus provides a baseline for assessing future

ECSW-augmented versions.

Declaration of generative AI and AI-assisted technologies in the writing process

During the preparation of this work the authors used chatGPT 4o in order to improve the readability in small sections of the article. After using this tool/service, the authors reviewed and edited the content as needed and take full responsibility for the content of the published article.

Appendix A. Deep Learning Implementation Details and Hyper-parameter Tuning

Appendix A.1. Overview

This appendix provides additional information on our deep learning implementation, focusing on hyper parameter tuning strategies, training procedures, and computational costs. These details aim to address fairness concerns in comparisons to other methods and to address model training time.

Appendix A.2. Hyperparameter Tuning for Our Model

We employed Bayesian search to explore hyperparameters in the limit of 300 runs per test case , ensuring appropriate architectures for both our method and the retrained concurrent approaches. Separate searches were performed for the Burgers equation, the cylinder flow, and the parametric cylinder flow. Table A.6 summarises the main hyperparameters considered.

Table A.6: Key hyperparameters tested in our grid searches.

Optimiser: {SGD, ADAM, AdaBelief}
Normalisation: {None, LayerNorm, BatchNorm}
Activation: {ReLU, LeakyReLU, SiLU}
Time Integrator (TI) hidden layers: [[64, 64, 64, 64], [128, 256, 128], [256, 128]]
Encoder/Decoder hidden layers: [[64], [128], [128, 64], [256, 128], [256, 96]]
Initial Learning Rate: {5e-3, 2e-3, 1e-3, 5e-4, 1e-4}
Learning Rate Decay: {Disabled, Enabled} (decayed to 1e-5)
Dropout in Time Integrator: {0.0, 0.1}
Reconstruction Loss: {L2 on POD coefs, L2 on full space, L2 + H1 norm}

For problems with a validation set (e.g. Burgers and parametric cylinder flows), training was halted once the validation loss stagnated. For the standard cylinder flow, we employed a fixed epoch budget of 20,000, after which all models converged or showed negligible improvement.

Concurrent Model Training. To ensure fair comparisons with fully data-driven and exclusively POD-based models, we preserved our chosen layer counts and widths for these concurrent architectures. Because our hybrid model adds relatively few parameters through its ROM-based encoding, this arrangement helped maintain a comparable parameter count among all methods. We retuned the remaining hyperparameters (e.g. optimiser, learning rate) for each model using the same grid search approach described above. Tuning of encoder / neural ODE / decoder fully data driven models in practice converged to the same optimal hyper parameter values to our models.

Appendix A.3. Training Protocol and Sequence Length Scheduling

All experiments were performed in PyTorch (v2.0.1) [68] with PyTorch Lightning (v2.1.2) [69], and tracked via Weights and Biases (v0.15.12) [70]. We used the Euler method within a Neural ODE framework for forward integration and no adjoint method for backpropagation, unless stated otherwise. Among SGD, ADAM, and AdaBelief, the latter emerged as a strong choice for most of our tests, though in specific settings (e.g. Burgers) ADAM sometimes performed comparably.

Sequence Length Scheduling. A key aspect of our training setup is the autoregressive nature of trajectory prediction, which can be computationally intensive. Instead of training on a long trajectory from the start, we progressively increased the sequence length as training epochs advanced. This helps prevent early divergence and allows the model to learn short-term dynamics before tackling longer time horizons. Below is a simplified outline of our scheduling approach:

- **Burgers equation**: Began with sub-sequences (25–50 time steps) and gradually incremented to a maximum of 300 steps by epoch 8,000.
- **Cylinder flow**: Initially trained on sequences of length 25; increased to 400 steps once the model stabilised (after 4,000 epochs).
- **Fluidic pinball**: Training began with 20-step sequences and doubled progressively (up to 80 steps) when loss dropped below 10^{-3}.

Appendix A.4. Training Times and Hardware Usage

Experiments were conducted on NVIDIA Tesla V100S GPUs. Table A.7 shows the best hyperparameters selected and wall clock average training times for each test case. Although the final architectures are relatively small, the sequential (autoregressive) nature of Neural ODE training can prolong training times, as it limits GPU parallelisation. Shortening the maximum sequence length or refining the scheduling strategy can substantially reduce training durations without severely degrading accuracy.

Table A.7: Selected best hyperparameters and approximate training times for each test case.

Hyperparameter	**Burgers**	**Cylinder**	**Param. Cylinder**	**Pinball Hybrid**
Optimiser	ADAM	AdaBelief	AdaBelief	AdamW
Normalisation (AE)	None	LayerNorm	LayerNorm	None
Normalisation (Time)	None	LayerNorm	LayerNorm	LayerNorm
Activation	LeakyReLU	SiLU	SiLU	SiLU
TI hidden layers	[256,128]	[128,256,128]	[128,256,128]	[256,256,256]
AE hidden layers	[256,96]	[256,128]	[256,128]	[256,128]
Max training epochs	10,000	20,000	20,000	20,000
Learning Rate Decay	Yes	Yes	Yes	Yes (exponential)
Approx. Training Time	$\sim$ 4 hours	$\sim$ 12 hours	$\sim$ 24 hours	$\sim$ 24 hours

In practice, once training is complete, the model's inference (also autoregressive) is significantly faster than high-fidelity simulations, offering speedups especially in multi-query contexts where repeated evaluations are required. Hence, while offline training can be lengthy, it can be justified by the substantial gains in computational efficiency during deployment.

Appendix B. Investigating Parameter Dependence of the Learnt Model for the Parametric Cylinder Flow Test Case

In this appendix, we analyse the parameter dependence of the cylinder-flow test case used in Sec. 7. Navier Stokes is a Non linear equation with respect to parameter choice (e.g., $\mathbf{u}(Re + \delta Re) \neq \mathbf{u}(Re) + \delta Re\, \partial\mathbf{u}/\partial Re$), our goal is to assess whether an affine-in-Re approximation is admissible over the parameter range considered. To this end, we quantify (i) the interpolation error of the steady base flow $\mathbf{w}_0(Re)$, and (ii) the impact on the dynamics prediction of using an affine hypothesis for the construction of the ROM. This evaluation clarifies the additional learning challenge induced by the parameter and motivates the inclusion of parametric tests.

Appendix B.1. Empirical Verification and Quantification of Non-Affine Behavior

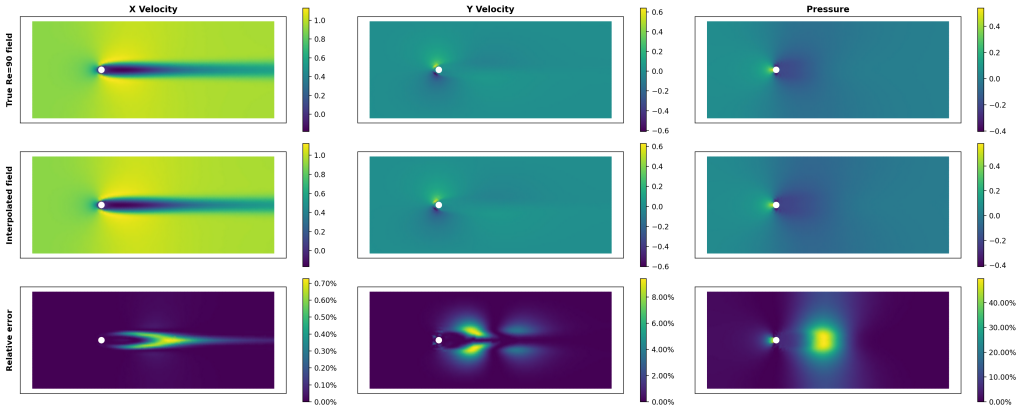


Figure B.28: Comparison between the true base flow at $Re = 90$ and the affine approximation $\mathbf{w}_{\text{lin}} = \frac{1}{2}(\mathbf{w}_{60} + \mathbf{w}_{120})$. The difference field highlights discrepancies near the end of the recirculation bubble.

Let $\mathbf{w} = [u, v, p]^\top$ denote the full state vector. We compare the base flow at $Re = 90$ ($\mathbf{w}_{90}$) with the linear approximation spanning the studied parameter range:

$$\mathbf{w}_{\text{lin}} := \frac{1}{2}\left(\mathbf{w}_{60} + \mathbf{w}_{120}\right).$$

The relative L^2 errors on the flow fields are $\sim 1\%$, $\sim 10\%$, and $\sim 40\%$ for u (streamwise velocity), v and p, respectively. Except for u, these values are far from negligible. Moreover, the largest discrepancies appear near the recirculation bubble (Figure B.28), which is known to be the zone that most strongly influences the unsteady dynamics in the cylinder flow case [71].

Appendix B.2. Impact on Derivative Predictions

Let us split the reduced state $\mathbf{a} \in \mathbb{R}^n$ ($n = 100$) as:

$$\mathbf{a} = \begin{bmatrix} \mathbf{a}_\parallel \\ \mathbf{a}_\perp \end{bmatrix}, \quad \mathbf{a}_\parallel \in \mathbb{R}^{n_\parallel = 3}, \quad \mathbf{a}_\perp \in \mathbb{R}^{n_\perp = 97}.$$

The Linearised Navier–Stokes operator can then be decomposed as:

$$\mathcal{L}(Re) = \begin{bmatrix} \mathcal{L}_{\parallel\parallel}(Re) & \mathcal{L}_{\parallel\perp}(Re) \\ \mathcal{L}_{\perp\parallel}(Re) & \mathcal{L}_{\perp\perp}(Re) \end{bmatrix}.$$

One can compare the true operator with its affine approximation, defined respectively as:

$$\mathcal{L}^{\text{true}} := \mathcal{L}(90), \quad \mathcal{L}^{\text{lin}} := \frac{1}{2}\mathcal{L}(60) + \frac{1}{2}\mathcal{L}(120).$$

To assess the impact of operator interpolation on dynamics, we compare instantaneous time derivatives predicted by the full reduced-order model, which includes the implicit linear term and explicit nonlinear term.

Nonlinear contribution..

$$[\mathcal{N}(\mathbf{a})]_i = \mathbf{a}^\top \mathcal{N}^{(i)}\, \mathbf{a}, \quad i = 1, \ldots, n, \tag{B.1}$$

where $\mathcal{N}^{(i)} \in \mathbb{R}^{n\times n}$ are the slices of the nonlinear tensor $\mathcal{N}$.

Time-stepping scheme.. The ROM is advanced using a second-order semi-implicit scheme:

$$\mathbf{a}^{n+1} = \mathbf{a}^n + \Delta t\, \dot{\mathbf{a}}^{n+1/2}, \tag{B.2}$$

with midpoint derivative:

$$\dot{\mathbf{a}}^{n+1/2} = \frac{4\mathbf{a}^n - \mathbf{a}^{n-1}}{2\Delta t} + 2\left(\mathcal{N}(\mathbf{a}^n) - \mathcal{N}(\mathbf{a}^{n-1})\right). \tag{B.3}$$

The derivative estimate is:

$$\mathbf{f} = \frac{1}{\Delta t}\left(\mathcal{M}^{-1}\mathbf{rhs} - \mathbf{a}^n\right), \tag{B.4}$$

where

$$\mathbf{rhs} = 4\mathbf{a}^n - \mathbf{a}^{n-1} + 4\Delta t\,\mathcal{N}(\mathbf{a}^n) - 2\Delta t\,\mathcal{N}(\mathbf{a}^{n-1}). \tag{B.5}$$

Comparison test.. For $(\mathbf{a}^{n-1}, \mathbf{a}^n)$ from the reference trajectory:

$$\mathbf{f}^{\text{true}} = \frac{1}{\Delta t}\left(\mathcal{M}_{\text{true}}^{-1}\mathbf{rhs} - \mathbf{a}^n\right), \tag{B.6}$$

$$\mathbf{f}^{\text{lin}} = \frac{1}{\Delta t}\left(\mathcal{M}_{\text{lin}}^{-1}\mathbf{rhs} - \mathbf{a}^n\right), \tag{B.7}$$

with $\mathcal{M}_{\text{true}} = I - \Delta t\,\mathcal{L}^{\text{true}}$ and $\mathcal{M}_{\text{lin}} = I - \Delta t\,\mathcal{L}^{\text{lin}}$.

Relative error:

$$\varepsilon = \frac{\|\mathbf{f}^{\text{true}} - \mathbf{f}^{\text{lin}}\|_2}{\|\mathbf{f}^{\text{true}}\|_2}. \tag{B.8}$$

Average errors:

$$\begin{aligned} \text{Full interpolation:} \quad & \varepsilon_{\parallel} = 3.67 \times 10^{-3}, \quad \varepsilon_{\perp} = 3.66 \times 10^{-2}, \\ \text{Only } \mathcal{L}_{\parallel\parallel} \text{ true:} \quad & \varepsilon_{\parallel} = 5.99 \times 10^{-4}, \quad \varepsilon_{\perp} = 3.66 \times 10^{-2}. \end{aligned}$$

Block-swap results:

$$\begin{aligned} \mathcal{L}_{\parallel\parallel} \text{ swapped: } & \varepsilon_{\parallel} = 4.40 \times 10^{-3}, \\ \mathcal{L}_{\parallel\perp} \text{ swapped: } & \varepsilon_{\parallel} = 8.89 \times 10^{-4}, \\ \mathcal{L}_{\perp\parallel} \text{ swapped: } & \varepsilon_{\perp} = 8.32 \times 10^{-3}, \\ \mathcal{L}_{\perp\perp} \text{ swapped: } & \varepsilon_{\perp} = 7.00 \times 10^{-3}. \end{aligned}$$

In conclusion, even with $\mathcal{L}_{\parallel\parallel}$ exact, coupling and $\perp$-blocks induce significant errors. A 5×10^{-4} error on $\parallel$ modes and 3×10^{-2} on $\perp$ modes implies decorrelation within $\mathcal{O}(100)$ steps for the latter and $\mathcal{O}(200)$ for the former (about one shedding cycle). This highlights the need for nonlinear parameter dependence modeling and justifies the test case as a nontrivial parametric example.

References

[1] C. Fefferman, Existence and smoothness of the navier-stokes equation, The Millennium Prize Problems (01 2006).

[2] X. I. Yang, K. P. Griffin, Grid-point and time-step requirements for direct numerical simulation and large-eddy simulation, Physics of Fluids 33 (1) (2021) 015108. doi:10.1063/5.0036515.

[3] K. Hami, Turbulence modeling a review for different used methods, International Journal of Heat and Technology 39 (1) (2021) 227–234. doi:10.18280/ijht.390125.

[4] K. Barkalov, I. Lebedev, M. Usova, D. Romanova, D. Ryazanov, S. Strijhak, Optimization of turbulence model parameters using the global search method combined with machine learning, Mathematics 10 (15) (2022) 2708. doi:10.3390/math10152708.

[5] G. Berkooz, P. Holmes, J. L. Lumley, The proper orthogonal decomposition in the analysis of turbulent flows, Annual review of fluid mechanics 25 (1) (1993) 539–575.

[6] N. Aubry, J. L. Lumley, The dynamics of coherent structures in the wall region of a turbulent boundary layer, Journal of Fluid Mechanics 192 (1988) 115 – 173, cited by: 1038. doi:10.1017/S0022112088001818.

[7] E. Gillies, Low-dimensional control of the circular cylinder wake, Journal of Fluid Mechanics 371 (1998) 157–178.

[8] B. R. Noack, K. Afanasiev, M. Morzyński, G. Tadmor, F. Thiele, A hierarchy of low-dimensional models for the transient and post-transient cylinder wake, Journal of Fluid Mechanics 497 (2003) 335–363.

[9] M. Couplet, C. Basdevant, P. Sagaut, Calibrated reduced-order pod-galerkin system for fluid flow modelling, Journal of Computational Physics 207 (1) (2005) 192–220.

[10] M. Bergmann, C.-H. Bruneau, A. Iollo, Enablers for robust pod models, Journal of Computational Physics 228 (2) (2009) 516–538.

[11] M. Bergmann, L. Cordier, J.-P. Brancher, Optimal rotary control of the cylinder wake using proper orthogonal decomposition reduced-order model, Physics of fluids 17 (9) (2005).

[12] A. Barbagallo, D. Sipp, P. J. Schmid, Closed-loop control of an open cavity flow using reduced-order models, Journal of Fluid Mechanics 641 (2009) 1–50.

[13] P. J. Schmid, Dynamic mode decomposition of numerical and experimental data, Journal of fluid mechanics 656 (2010) 5–28.

[14] A. Alomar, A. Nicole, D. Sipp, V. Rialland, F. Vuillot, Reduced-order model of a reacting, turbulent supersonic jet based on proper orthogonal decomposition, Theoretical and Computational Fluid Dynamics 34 (1) (2020) 49–77.

[15] S. E. Ahmed, O. San, A. Rasheed, T. Iliescu, Nonlinear proper orthogonal decomposition for convection-dominated flows, Physics of Fluids 33 (12) (2021) 121702. doi:10.1063/5.0074310.

[16] F. Romor, G. Stabile, G. Rozza, Non-linear manifold rom with convolutional autoencoders and reduced over-collocation method (2022). arXiv:2203.00360.

[17] F. Pichi, B. Moya, J. S. Hesthaven, A graph convolutional autoencoder approach to model order reduction for parametrized pdes (2023). arXiv:2305.08573.

[18] F. J. Gonzalez, M. Balajewicz, Deep convolutional recurrent autoencoders for learning low-dimensional feature dynamics of fluid systems (2018). arXiv:1808.01346.

[19] A. Olmo, A. Zamzam, A. Glaws, R. King, Physics-driven convolutional autoencoder approach for cfd data compressions (2022). arXiv:2210.09262.
URL `https://arxiv.org/abs/2210.09262`

[20] G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, L. Yang, Physics-informed machine learning, Nature Reviews Physics 3 (6) (2021) 422–440. doi:10.1038/s42254-021-00314-5.

[21] R. Zwanzig, Nonequilibrium Statistical Mechanics, Oxford University Press., 2000.

[22] Q. Wang, N. Ripamonti, J. S. Hesthaven, Recurrent neural network closure of parametric pod-galerkin reduced-order models based on the mori-zwanzig formalism, Journal of Computational Physics 410 (2020) 109402. doi:10.1016/j.jcp.2020.109402.

[23] E. Menier, M. A. Bucci, M. Yagoubi, L. Mathelin, M. Schoenauer, Cd-rom: Complemented deep - reduced order model, Computer Methods in Applied Mechanics and Engineering 410 (2023) 115985. doi:10.1016/j.cma.2023.115985.

[24] P. Wu, J. Sun, X. Chang, W. Zhang, R. Arcucci, Y. Guo, C. C. Pain, Data-driven reduced order model with temporal convolutional neural network, Computer Methods in Applied Mechanics and Engineering 360 (2020) 112766. doi:10.1016/j.cma.2019.112766.

[25] Z. Wang, I. Akhtar, J. Borggaard, T. Iliescu, Proper orthogonal decomposition closure models for turbulent flows: A numerical comparison, Computer Methods in Applied Mechanics and Engineering 237–240 (2012) 10–26. doi:10.1016/j.cma.2012.04.015.

[26] A. Ivagnes, G. Stabile, A. Mola, T. Iliescu, G. Rozza, Pressure data-driven variational multiscale reduced order models, SSRN Electronic Journal (2022). doi:10.2139/ssrn.4134905.

[27] Y. Iwasaki, T. Nagata, Y. Sasaki, K. Nakai, M. Inubushi, T. Nonomura, Reservoir computing reduced-order model based on particle image velocimetry data of post-stall flow, AIP Advances 13 (6) (Jun 2023). doi:10.1063/5.0150947.

[28] J. L. Barnett, C. Farhat, Y. Maday, Neural-network-augmented projection-based model order reduction for mitigating the kolmogorov barrier to reducibility of cfd models (Dec 2022).
URL https://arxiv.org/abs/2212.08939

[29] P. Gupta, P. J. Schmid, D. Sipp, T. Sayadi, G. Rigas, Mori-zwanzig latent space koopman closure for nonlinear autoencoder, arXiv preprint arXiv:2310.10745 (2023).

[30] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436–444. doi:10.1038/nature14539.

[31] K. Lee, K. T. Carlberg, Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders, Journal of Computational Physics 404 (2020) 108973. doi:10.1016/j.jcp.2019.108973.

[32] F. Romor, G. Stabile, G. Rozza, Non-linear manifold reduced-order models with convolutional autoencoders and reduced over-collocation method, Journal of Scientific Computing 94 (3) (Feb 2023). doi:10.1007/s10915-023-02128-2.

[33] F. Romor, G. Stabile, G. Rozza, Explicable hyper-reduced order models on nonlinearly approximated solution manifolds of compressible and incompressible navier-stokes equations, Journal of Computational Physics 524 (2025) 113729. doi:10.1016/j.jcp.2025.113729.

[34] Y. Kim, Y. Choi, D. Widemann, T. Zohdi, A fast and accurate physics-informed neural network reduced order model with shallow masked autoencoder, Journal of Computational Physics 451 (2022) 110841. doi:10.1016/j.jcp.2021.110841.

[35] B. Lusch, J. N. Kutz, S. L. Brunton, Deep learning for universal linear embeddings of nonlinear dynamics, Nature Communications 9 (1) (Nov 2018). doi:10.1038/s41467-018-07210-0.

[36] R. Halder, K. J. Fidkowski, K. J. Maki, Non-intrusive reduced-order modeling using convolutional autoencoders, International Journal for Numerical Methods in Engineering 123 (21) (2022) 5369–5390. doi:10.1002/nme.7072.

[37] A. S. Nair, S. Barwey, P. Pal, J. F. MacArt, T. Arcomano, R. Maulik, Understanding latent timescales in neural ordinary differential equation models for advection-dominated dynamical systems (2025). arXiv:2403.02224.
URL `https://arxiv.org/abs/2403.02224`

[38] T. Simpson, N. Dervilis, E. Chatzi, Machine learning approach to model order reduction of nonlinear systems via autoencoder and LSTM networks, J. Eng. Mech. 147 (10) (2021) 04021061.

[39] A. Beiki, R. Kamali, Novel attention-based convolutional autoencoder and convlstm for reduced-order modeling in fluid mechanics with time derivative architecture, Physica D: Nonlinear Phenomena 454 (2023) 133857. doi:10.1016/j.physd.2023.133857.

[40] S. Fresca, A. Manzoni, Pod-dl-rom: Enhancing deep learning-based reduced order models for nonlinear parametrized pdes by proper orthogonal decomposition, Computer Methods in Applied Mechanics and Engineering 388 (2022) 114181. doi:10.1016/j.cma.2021.114181.

[41] M. Barrault, Y. Maday, N. C. Nguyen, A. T. Patera, An 'empirical interpolation' method: Application to efficient reduced-basis discretization of partial differential equations, Comptes Rendus. Mathématique 339 (9) (2004) 667–672. doi:10.1016/j.crma.2004.08.006.

[42] S. Chaturantabut, D. C. Sorensen, Nonlinear model reduction via discrete empirical interpolation, SIAM Journal on Scientific Computing 32 (5) (2010) 2737–2764. doi:10.1137/090766498.

[43] K. Carlberg, Y. Choi, S. Sargsyan, Conservative model reduction for finite-volume models, Journal of Computational Physics 371 (2018) 280–314. doi:10.1016/j.jcp.2018.05.019.

[44] A. Iollo, S. Lanteri, J.-A. Désidéri, Stability properties of pod-galerkin approximations for the compressible navier-stokes equations, Theoretical and Computational Fluid Dynamics 13 (6) (2000) 377–396. doi:10.1007/s001620050119.

[45] E. Qian, B. Kramer, B. Peherstorfer, K. Willcox, Lift & learn: Physics-informed machine learning for large-scale nonlinear dynamical systems, Physica D: Nonlinear Phenomena 406 (2020) 132401. doi:https://doi.org/10.1016/j.physd.2020.132401.
URL https://www.sciencedirect.com/science/article/pii/S0167278919307651

[46] R. T. Q. Chen, Y. Rubanova, J. Bettencourt, D. Duvenaud, Neural ordinary differential equations, NeurIPS (2018). doi:https://doi.org/10.48550/arXiv.1806.07366.

[47] R. Li, X. Wang, G. Huang, W. Yang, K. Zhang, X. Gu, S. Tran, S. Garg, J. Alty, Q. Bai, A comprehensive review on deep supervision: Theories and applications, ArXiv abs/2207.02376 (2022).
URL https://api.semanticscholar.org/CorpusID:250311644

[48] A. J. Linot, J. W. Burby, Q. Tang, P. Balaprakash, M. D. Graham, R. Maulik, Stabilized neural ordinary differential equations for long-time forecasting of dynamical systems, ArXiv abs/2203.15706 (2022).
URL https://api.semanticscholar.org/CorpusID:247779034

[49] J. Zhuang, N. Dvornek, X. Li, S. Tatikonda, X. Papademetris, J. Duncan, Adaptive checkpoint adjoint method for gradient estimation in neural ODE, in: H. D. III, A. Singh (Eds.), Proceedings of the 37th International Conference on Machine Learning, Vol. 119 of Proceedings of Machine Learning Research, PMLR, 2020, pp. 11639–11649.
URL https://proceedings.mlr.press/v119/zhuang20a.html

[50] J. BEC, K. KHANIN, Burgers turbulence, Physics Reports 447 (1–2) (2007) 1–66. doi:10.1016/j.physrep.2007.04.002.
URL http://dx.doi.org/10.1016/j.physrep.2007.04.002

[51] J. M. Burgers, The nonlinear diffusion equation: Asymptotic Solutions and statistical problems, Springer Netherlands, 2013.

[52] E. Hopf, The partial differential equation $u_t + uu_x = \mu u_{xx}$, Communications on Pure and Applied Mathematics 3 (3) (1950) 201–230. doi:10.1002/cpa.3160030302.

[53] M. W. Scroggs, J. S. Dokken, C. N. Richardson, G. N. Wells, Construction of arbitrary order finite element degree-of-freedom maps on polygonal and polyhedral cell meshes, ACM Transactions on Mathematical Software 48 (2) (2022) 1–23. doi:10.1145/3524456.
URL `https://doi.org/10.1145%2F3524456`

[54] M. S. Alnaes, A. Logg, K. B. Oelgaard, M. E. Rognes, G. N. Wells, Unified form language: A domain-specific language for weak formulations of partial differential equations (2013). arXiv:1211.4047.

[55] K. Perlin, An image synthesizer, SIGGRAPH Comput. Graph. 19 (3) (1985) 287–296. doi:10.1145/325165.325247.

[56] C. H. K. Williamson, Vortex dynamics in the cylinder wake, Annual Review of Fluid Mechanics 28 (Volume 28, 1996) (1996) 477–539. doi:https://doi.org/10.1146/annurev.fl.28.010196.002401.

[57] B. Noack, K. Afanasiev, M. Morzynski, G. Tadmor, F. Thiele, A hierarchy of low-dimensional models for the transient and post-transient cylinder wake, Journal of Fluid Mechanics 497 (2003) 335–363. doi:10.1017/S0022112003006694.

[58] G. Tadmor, O. Lehmann, B. Noack, M. Morzynski, Mean field representation of the natural and actuated cylinder wake, Physics of Fluids - PHYS FLUIDS 22 (03 2010). doi:10.1063/1.3298960.

[59] F. Hecht, New development in freefem++, J. Numer. Math. 20 (3-4) (2012) 251–265.
URL `https://freefem.org/`

[60] N. Deng, B. R. Noack, M. Morzyński, L. R. Pastur, Low-order model for successive bifurcations of the fluidic pinball, Journal of Fluid Mechanics 884 (Dec 2019). doi:10.1017/jfm.2019.959.

[61] B. Fröhlich, J. Gade, F. Geiger, M. Bischoff, P. Eberhard, Geometric element parameterization and parametric model order reduction in finite element based shape optimization, Computational Mechanics 63 (5) (2018) 853–868. doi:10.1007/s00466-018-1626-1.

[62] R. Maulik, V. Rao, J. Wang, G. Mengaldo, E. Constantinescu, B. Lusch, P. Balaprakash, I. Foster, R. Kotamarthi, Efficient high-dimensional variational data assimilation with machine-learned reduced-order models (2021). arXiv:2112.07856.
URL https://arxiv.org/abs/2112.07856

[63] P. Kumari, B. Bhadriraju, Q. Wang, J. S.-I. Kwon, Development of parametric reduced-order model for consequence estimation of rare events, Chemical Engineering Research and Design 169 (2021) 142–152. doi:https://doi.org/10.1016/j.cherd.2021.02.006.

[64] S. Georgaka, G. Stabile, G. Rozza, M. J. Bluck, Parametric pod-galerkin model order reduction for unsteady-state heat transfer problems (2019). arXiv:1808.05175.
URL https://arxiv.org/abs/1808.05175

[65] D. Amsallem, J. Cortial, K. Carlberg, C. Farhat, A method for interpolating on manifolds structural dynamics reduced-order models, International Journal for Numerical Methods in Engineering 80 (9) (2009) 1241–1258. doi:10.1002/nme.2681.

[66] D. P. Kingma, D. J. Rezende, S. Mohamed, M. Welling, Semi-supervised learning with deep generative models (2014). arXiv:1406.5298.
URL https://arxiv.org/abs/1406.5298

[67] K. Sohn, H. Lee, X. Yan, Learning structured output representation using deep conditional generative models, in: C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, R. Garnett (Eds.), Advances in Neural Information Processing Systems, Vol. 28, Curran Associates, Inc., 2015.

[68] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, S. Chintala, Pytorch: An imperative style, high-performance deep learning library, in: Advances in Neural Information Processing Systems 32, Curran Associates, Inc., 2019, pp. 8024–8035.

[69] W. Falcon, T. P. L. team, Pytorch lightning, version 1.4. Available at: https://github.com/Lightning-AI/lightning (2019). doi:10.5281/zenodo.3828935.

[70] L. Biewald, Experiment tracking with weights and biases, software available from wandb.com (2020).
URL https://www.wandb.com/

[71] O. Marquet, D. Sipp, L. Jacquin, Sensitivity analysis and passive control of cylinder flow, Journal of Fluid Mechanics 615 (2008) 221–252. doi:10.1017/s0022112008003662.