

UNIVERSIDAD POLITÉCNICA DE MADRID
Escuela Técnica Superior de Ingeniería Aeronáutica y del Espacio



Deep Learning and Modal Decomposition for Reduced-Order Modeling of High-Dimensional Flow Dynamics

DOCTORAL THESIS

Submitted for the degree of Doctor by:

Santos Rodrigo Abadía Heredia
Máster en Matemática Industrial (M2i)

Madrid, 2025



UNIVERSIDAD POLITÉCNICA DE MADRID
Escuela Técnica Superior de Ingeniería Aeronáutica y del
Espacio

Doctoral Degree in Aerospace Engineering

Deep Learning and Modal Decomposition for Reduced-Order Modeling of High-Dimensional Flow Dynamics

DOCTORAL THESIS

Submitted for the degree of Doctor by:

Santos Rodrigo Abadía Heredia

Máster en Matemática Industrial (M2i)

Under the supervision of:

Dr. Soledad Le Clainche Martínez

Dr. Manuel López Martín

Madrid, 2025

Title: Deep Learning and Modal Decomposition for Reduced-Order Modeling of High-Dimensional Flow Dynamics

Author: Santos Rodrigo Abadía Heredia

Doctoral Programme: Aerospace Engineering

Thesis Supervision:

Dr. Soledad Le Clainche Martínez, Full Professor, Escuela Técnica Superior de Ingeniería Aeronautica y de Espacio, Universidad Politécnica de Madrid.

Dr. Manuel López Martín, Senior Deep Learning Scientist, Advanced Mathematics Group, Repsol.

External Reviewers:

Dr. Julio Soria, Full Professor, Monash University.

Dr. Marcello Meldi, Full Professor, École Nationale Supérieure d'Arts et Métiers.

Thesis Defense Committee:

Thesis Defense Date:

This thesis has been partially supported by the grants MODEL-CO, PID2020-114173RB-I00, TED2021-129774B-C21 and PLEC2022-009235 funded by the Comunidad de Madrid y Ministerio de Ciencia e Innovación, Spain. Also, Rodrigo Abadía Heredia and Soledad Le Clainche Martínez acknowledge the support of Programa Propio I+D+i 2023 and the Programa de realización de estancias de investigación en universidades de E.E.U.U (2024/25) funded by the Universidad Politécnica de Madrid, which funded two international research stays during the PhD.

*Get along on down the road
We've got a long, long way to go*

Agradecimientos

Al final de esta tesis doctoral quisiera dedicar un momento a agradecer a las personas que me han acompañado durante este proceso y han permitido que se haga real.

En primer lugar quisiera agradecer a mi directora de tesis Soledad Le Clainche, por su orientación, dedicación y el esfuerzo extra que ha hecho todos estos años por buscarme financiación y las dos estancias. Esta tesis no se habría podido realizar sin su apoyo y ayuda, ya que ha sido el nexo principal entre el campo del machine learning y la mecánica de fluidos, dando origen a esta tesis.

Quiero también agradecer a mi codirector Manuel López Martín, por su orientación en el campo de la inteligencia artificial, ya que fue él quien hizo que me interesara por los modelos generativos que en su momento acababan de aparecer. Las conversaciones tenidas con él siempre han llevado a que siempre aprendiera algo nuevo sobre el campo.

Quiero dar las gracias a Alessandro Parente, quien me aceptó en su grupo de investigación en mi primera estancia en el extranjero, los aprendizajes obtenidos dentro de su grupo durante este periodo han sido de gran valor para mí durante la realización de la tesis.

También agradezco a Petros Koumoutsakos, quien me aceptó en su grupo de investigación durante mi segunda estancia en el extranjero, las enseñanzas obtenidas tanto por él como durante la estancia han sido de gran valor tanto para la tesis como en lo personal.

A mi familia, por su apoyo durante estos cinco años de tesis, dandome el cariño y el arropo que han permitido poder seguir adelante durante este camino.

A mis compañeros de doctorado, por su guía, amistad y compañía que han permitido hacer que la tesis sea más disfrutable y llevadera, y que sin darnos cuenta nos llevamos viendo las caras casi todos los días desde hace ya varios años.

Quiero finalmente agradecer a María, quien ha sido uno de los pilares claves que han permitido que esta tesis pudiese llegar a una conclusión, su ayuda y apoyo durante nuestras múltiples conversaciones han sido de gran importancia par mí.

Esta tesis ha podido salir adelante gracias a muchas personas que han aportado su granito de arena a la culminación de este proyecto.

Rodrigo Abadía Heredia
Julio de 2025

Abstract

This thesis focuses on the development of reduced-order models that combine modal decomposition techniques with deep learning (DL) architectures to predict the dynamics of fluid systems at different Reynolds numbers. The use of artificial intelligence (AI)-based models to model physical systems represents a currently active field in the academic community with broad industrial applications.

The motivation for studying these models is based on two main advantages: their highly nonlinear capacity to identify complex patterns within structured and unstructured datasets, and their low computational cost for inferring predictions after training. These characteristics are particularly attractive to the fluid mechanics community, which studies highly complex flows defined by multiple coherent structures at different spatio-temporal scales, difficult to capture and computationally expensive, with numerical simulation being unfeasible in some cases.

However, these models require a rich and varied dataset for training, whose acquisition constitutes the main obstacle in fluid dynamics, where accurate data can only be obtained through computationally expensive numerical simulations or experiments with high monetary cost. Furthermore, each sample has a high spatial dimensionality, requiring complex DL architectures.

To address these challenges, a hybrid methodology that combines modal decomposition with DL architectures was studied, where modal decomposition reduces the dimensionality of the samples, allowing the use of simpler architectures and reduced datasets, consuming fewer computational resources for training than purely DL-based models.

Since AI-based models tend to generate predictions that rapidly diverge from the target dynamics, an adaptive methodology was developed that leverages the advantages of the hybrid model. This allows initial training on high-fidelity numerical simulation data, obtained using reliable but expensive classical numerical methods, to predict the temporal evolution up to a certain point, and then simulate numerically again to retrain the model with new information, continuously updating it with fresh data about the flow dynamics.

Finally, a generative AI model was studied that, contrary to the hybrid model, captures the small-scale structures characteristic of turbulent flow dynamics. This model also uses dimensional reduction techniques to train on a simplified representation, requiring more data than the hybrid model but less than similar DL models.

The results demonstrate that AI-based models can be widely applied in fluid dynamics problems, allowing prediction of the temporal evolution of flows without the high computational cost of current numerical simulation techniques, offering an efficient alternative for predictive modeling of complex fluid systems.

Resumen

Esta tesis se centra en el desarrollo de modelos de orden reducido que combinan técnicas de descomposición modal con arquitecturas de deep learning (DL) para predecir la dinámica de sistemas fluidos a distintos números de Reynolds. El uso de modelos basados en inteligencia artificial (IA) para modelizar sistemas físicos representa un campo actualmente activo en la comunidad académica con amplias aplicaciones industriales.

La motivación para estudiar estos modelos se fundamenta en dos ventajas principales: su capacidad altamente no lineal para identificar patrones complejos dentro de bases de datos estructuradas y no estructuradas, y su bajo coste computacional para inferir predicciones tras el entrenamiento. Estas características resultan especialmente atractivas para la comunidad de mecánica de fluidos, que estudia flujos altamente complejos definidos por múltiples estructuras coherentes a distintas escalas espaciotemporales, difíciles de capturar y computacionalmente costosos, siendo en algunos casos inviable una simulación numérica.

Sin embargo, estos modelos requieren de una rica y variada base de datos para su entrenamiento, cuya obtención constituye el principal obstáculo en mecánica de fluidos. Los datos precisos solo se consiguen mediante simulaciones numéricas computacionalmente costosas o experimentos con alto coste monetario. Además, cada muestra tiene una alta dimensión espacial, necesitando arquitecturas de DL complejas.

Para abordar estos desafíos, se estudió una metodología híbrida que combina descomposición modal con arquitecturas de DL. La descomposición modal reduce la dimensionalidad de las muestras, permitiendo usar arquitecturas más simples y bases de datos reducidas, consumiendo menos recursos computacionales para su entrenamiento que modelos puramente basados en DL.

Dado que los modelos basados en IA suelen generar predicciones que divergen rápidamente de la dinámica objetivo, se desarrolló una metodología adaptativa que aprovecha las ventajas del modelo híbrido. Esta permite entrenar inicialmente sobre datos de simulación numérica, usando métodos clásicos fiables pero costosos, para predecir la evolución temporal hasta cierto punto, y luego volver a simular numéricamente para reentrenar el modelo con nueva información, actualizándolo continuamente con datos frescos sobre la dinámica del flujo.

Finalmente, se estudió un modelo de IA generativa que, contrariamente al modelo híbrido, captura las escalas más pequeñas de la dinámica del flujo. Este modelo también utiliza técnicas de reducción dimensional para entrenar en una representación simplificada, requiriendo más datos que el modelo híbrido pero menos que modelos DL similares.

Los resultados demuestran que los modelos basados en IA pueden aplicarse ampliamente en mecánica de fluidos, permitiendo predecir la evolución temporal de flujos sin el alto coste computacional de las técnicas actuales de simulación numérica, ofreciendo una alternativa eficiente para el modelado predictivo de sistemas fluidos complejos.

Table of Contents

Acknowledgement	v
Abstract	vi
Resumen	vii
List of Figures	x
List of Tables	xviii
Abbreviations and acronyms	xxi
1 Introduction	1
1.1 Research contributions and novelty	7
1.2 Publications published during the thesis, presentations at conferences and patent	8
1.2.1 Publications	8
1.2.2 Presentations at conferences	9
1.2.3 Patent	10
2 The POD-DL: A hybrid methodology combining modal decomposition and deep learning for flow dynamics forecasting.	11
2.1 Projection-based ROM	11
2.2 The POD-DL methodology	13
2.2.1 Implementation of POD	14
2.2.2 Forecasting DL model	16
2.3 Results	17
2.3.1 Three-dimensional wake of a flow past a circular cylinder.	21
2.4 Discussion	26
3 Generalization capabilities of the hybrid POD-DL model against fully DL models	33
3.1 Methodology	34
3.1.1 Point forecasting: POD-DL and residual convolutional autoencoder (RAE)	35
3.1.2 Probabilistic forecasting: Variational autoencoder (VAE)	38
3.2 Results	40
3.2.1 Three-dimensional wake behind a circular cylinder: laminar regime .	42
3.2.2 Three-dimensional wake behind a circular cylinder: turbulent regime	46
3.2.3 Stability of POD-DL at different numbers of training samples	53
3.2.4 Results from a pre-trained model	55

3.3	Discussion	58
4	An adaptive framework for autoregressive forecasting in CFD using the POD-DL model	61
4.1	Methods: adaptive framework	62
4.2	Forecasting deep learning model	64
4.3	Implementation of the adaptive framework	66
4.4	Results	69
4.4.1	Three-dimensional wake behind a circular cylinder at laminar regime: laminar flow	69
4.4.2	Three-dimensional wake behind a circular cylinder at turbulent regime: turbulent flow	73
4.4.3	Isothermal subsonic jet	78
4.5	Discussion	83
5	Generative AI for capturing small-scale structures in turbulence	85
5.1	Method: G-LED	86
5.1.1	Forecasting model: Transformer	87
5.1.2	Recovering of micro dynamics: generative model	88
5.2	Implementation of G-LED within the POD-DL framework	90
5.3	Results	90
5.3.1	Results from G-LED without POD	91
5.3.2	Results from G-LED combined with POD	94
5.4	Discussion	98
6	Conclusions	101
6.1	Advances in modeling high-dimensional flows with POD-DL	102
6.2	Limitations, challenges and future work	104
	References	107
A	Data Preprocessing	119
A.1	Data augmentation	120
A.2	Rolling window	121
B	Computation of the probability of prediction error (PPE)	123
C	Model architectures and analysis of autoencoder latent spaces	125

List of Figures

2.1	The POD-DL framework.	14
2.2	Temporal evolution of the spatially averaged streamwise (a) and normal (b) velocity components. Here, t_s denotes the snapshot index rather than the actual time t	22
2.3	Top row shows the singular value decay for $\text{Norm}_1 = \text{None}$ (a), $\text{Norm}_1 = \text{Range}$ (b) and $\text{Norm}_1 = \text{Standardization}$ (c). Bottom row shows the cummulative energy for $\text{Norm}_1 = \text{None}$ (d), $\text{Norm}_1 = \text{Range}$ (e) and $\text{Norm}_1 = \text{Standardization}$ (f).	24
2.4	The top row shows the comparison of the first (γ_1 ; a), second (γ_2 ; b) and fourth (γ_4 ; c) POD coefficients derived from ground truth data (computed according to eq. (2.29)) and obtained from POD-DL predictions across different Norm_2 scaling strategies, with $\text{Norm}_1 = \text{None}$. The middle row shows the corresponding first (d), second (e) and fourth (f) coefficients when $\text{Norm}_1 = \text{Range}$. The bottom row displays the first (g), second (h) and fourth (i) coefficients for $\text{Norm}_1 = \text{Standardization}$	25
2.5	The top row shows the RMSE (a) and SSIM (b) obtained across different Norm_2 scaling approaches, with $\text{Norm}_1 = \text{None}$. The middle row shows the corresponding RMSE (c) and SSIM (d) results when $\text{Norm}_1 = \text{Range}$. The bottom row displays the RMSE (e) and SSIM (f) metrics for $\text{Norm}_1 = \text{Standardization}$	27
2.6	Comparison of ground truth (left column) and predicted (middle column) vorticity field for snapshots 50 (a) and 190 (b), with $\text{Norm}_1 = \text{None}$ and $\text{Norm}_2 = \text{None}$. The right column in both figures shows the absolute error between the predicted and ground truth vorticity.	28
2.7	Comparison of ground truth (left column) and predicted (middle column) vorticity field for snapshots 50 (a) and 190 (b), with $\text{Norm}_1 = \text{Range}$ and $\text{Norm}_2 = \text{Standardization}$. The right column in both figures shows the absolute error between the predicted and ground truth vorticity.	29
2.8	Comparison of ground truth (left column) and predicted (middle column) vorticity field for snapshots 50 (a) and 190 (b), with $\text{Norm}_1 = \text{Standardization}$ and $\text{Norm}_2 = \text{Range}$. The right column in both figures shows the absolute error between the predicted and ground truth vorticity.	30

3.1	Visual representation of the two residual block types used in the RAE: identity block (top) and convolutional block (bottom). Both structures implement residual connections by summing the block's input with its output. The key distinction lies in their effect on spatial dimensions, the identity block preserves the input resolution, whereas the convolutional block reduces it through downsampling.	36
3.2	Overview of the residual autoencoder (RAE) architecture. Note, in contrast to the POD-DL model, both the encoder and decoder components of this architecture are constructed entirely from CNNs.	37
3.3	Overview of the VAE architecture. Similar to the residual autoencoder, the encoder and decoder of this model are composed of CNNs.	38
3.4	(a) Streamwise and (b) wall-normal velocity evolution in the laminar flow. Black, blue and orange lines represent the transitory state of the flow, the portion of data used for training and the portion used for evaluating the models, respectively.	43
3.5	(a) Mode decay and (b) cumulative energy for the 40 most energetic modes in the laminar flow.	43
3.6	Snapshots of the streamwise velocity field from the laminar flow at time instants (a) 450 and (b) 485, highlighting in red points the spatial locations P_1 (left) and P_2 (right). These reference points are used to evaluate the temporal evolution of the streamwise ($\mathbf{v}_x$) and wall-normal ($\mathbf{v}_y$) velocity components, as presented in Fig. 3.7 and Fig. 3.8.	44
3.7	Comparison of the ground truth streamwise velocity evolution $u_x(t)$, at spatial coordinate P_1 shown in Fig. 3.6, against the predictions obtained from the (a) POD-DL, (b) residual autoencoder and (c) variational autoencoder models, respectively. Counter part for the spatial coordinate P_2 in figures (d), (e) and (f). In all figures the ground truth is represented by a solid line and the prediction by a dashed line.	44
3.8	Same as Fig. 3.7 for the wall-normal velocity, $\mathbf{v}_y(t_S)$	45
3.9	Prediction error in the laminar flow scenario as quantified by (a) the RMSE for the streamwise velocity component, (b) the RMSE for the wall-normal component, (c) the SSIM for the streamwise component and (d) the SSIM for the wall-normal component. These metrics evaluate the performance of each model in capturing the temporal dynamics and structural fidelity of the velocity fields.	46
3.10	Representative snapshots of the streamwise velocity component from the laminar flow at time instants 450, 500, and 550, displaying the ground truth, model predictions and corresponding absolute error maps in grayscale. Results are shown for each of the evaluated models to highlight predictive accuracy and spatial error distribution over time.	47
3.11	Representative snapshots of the wall-normal velocity component from the laminar flow at time instants 450, 500, and 550, displaying the ground truth, model predictions and corresponding absolute error maps in grayscale. Results are shown for each of the evaluated models to highlight predictive accuracy and spatial error distribution over time.	48

3.12	Temporal evolution of the spatially averaged streamwise velocity, $\bar{v}_x$, in the turbulent flow. The blue line indicates the portion of the data used for model training, while the orange line represents the unseen dynamics from where the first 200 snapshots are used for evaluating the models.	49
3.13	(a) Mode decay and (b) cumulative energy for the 100 most energetic modes in the turbulent flow.	50
3.14	Comparison of the original flow from experimental measurements (left column) and the reconstruction using SVD (right column), keeping the 20 most energetic modes to reconstruct the flow field.	50
3.15	Snapshots from the turbulent flow streamwise velocity, at time instants (a) 2800 and (b) 2876, representing by red points, P_1 (left) and P_2 (right), the spatial coordinates used to compare the streamwise velocity evolution shown in Fig. 3.16.	51
3.16	Comparison of the ground truth turbulent flow streamwise velocity evolution at the spatial coordinate P_1 shown in Fig. 3.15, against the predictions obtained from the (a) POD-DL, (c) RAE and (e) VAE, respectively. Counter part for the spatial coordinate P_2 in figures (b), (d) and (f). In all figures the ground truth is represented by a solid line and the prediction by a dashed line. . . .	52
3.17	Error committed by the models, in each prediction for the streamwise velocity component, measured by the (a) RRMSE and (b) the SSIM.	52
3.18	Snapshots at some representative time instants showing the ground truth streamwise velocity component of the turbulent flow, the predictions obtained from the different models and their respective absolute error in gray-scale. . .	53
3.19	Comparison of the probability density function between the ground truth and predicted velocity field of the turbulent flow obtained from (a) POD-DL, (b) RAE and (c) VAE models, respectively.	54
3.20	Prediction error of the POD-DL model trained with different number of samples (indicated in brackets) for the streamwise velocity component of the turbulent flow past a cylinder. Errors are quantified using (a) the RMSE and (b) the SSIM defined in (2.22) and (2.23), respectively.	54
3.21	Snapshots at some representative time instants showing the ground truth streamwise velocity component of the turbulent flow, the predictions obtained from the POD-DL, trained with different training data lengths, and their respective absolute error in gray-scale.	55
3.22	Comparison of the original flow from the numerical simulation of the Jet LES (left column) and the reconstruction using SVD (right column), retaining the 20 most energetic modes to reconstruct the flow field, at two different time steps (top and bottom row, respectively).	56
3.23	Error committed by the POD-DL model with and without retraining, on the streamwise velocity component of the Jet LES. Errors are quantified using the SSIM defined in (2.23).	56
3.24	Comparison of the velocity histograms between the ground truth and the temporal predictions of the Jet LES obtained from (a) the POD-DL without retraining and (b) the POD-DL with retraining, respectively.	57

3.25	Snapshots at some representative time instants showing the ground truth streamwise velocity (left column) and the predictions, from the POD-DL model, for the Jet LES. No retrain (middle column) makes reference to the predictions obtain from case (1) and Retrain (right column) from case (2), described at the beginning of Section 3.2.4.	58
4.1	(a) Evolution of the learning rate over 1500 epochs under the CosineAnnealing-WarmRestarts schedule, oscillating between 10^{-3} and a minimum value of 10^{-6} . (b) Corresponding training loss, showing pronounced decreases immediately after each warm restart as the learning rate returns to its peak value.	66
4.2	Schematic representation of the adaptive framework proposed in this chapter. Initially, S_0 snapshots are extracted from a precomputed dataset to train the POD-DL model. POD is applied to obtain the corresponding coefficients, which are then used to train a DL model over E epochs. The trained model forecasts P future snapshots, after which an additional set of S_1 snapshots is retrieved for retraining. During each retraining cycle, the DL model employs transfer learning by reusing the weights from the previous training phase. This process is iteratively repeated until predictions span the entire temporal domain of interest.	67
4.3	Percentage of cumulative energy, $E(k)$, retained when truncating at the k most energetic modes (up to $k = 20$) for the laminar flow.	70
4.4	Comparison of ground truth (GT) and POD-DL predicted vorticity fields for the laminar flow at representative time steps from test cases T1 (a), T2 (b) and T3 (c). The index t_S refers to the snapshot index, not the physical time t	71
4.5	The top row shows the probability of achieving an absolute vorticity prediction error below thresholds of 0.5 (orange) and 1 (green) for the three test cases: T1 (a), T2 (b) and T3 (c), in the laminar flow. The bottom row illustrates the temporal evolution of the spatially averaged streamwise velocity. The selected error thresholds are based on the typical vorticity range observed in the flow, as indicated in Fig. 4.4.	72
4.6	Temporal evolution of the 0.25-th (bottom), 0.5-th (middle) and 0.75-th (top) vorticity percentiles for the ground truth (blue) and POD-DL predictions (orange) across test cases T1 (a), T2 (b) and T3 (c) in the laminar flow scenario. Shaded regions denote the time intervals corresponding to snapshots used during both initial training and subsequent retraining phases.	73
4.7	Cumulative energy percentage, $E(k)$, retained when truncating at the k most energetic modes (up to $k = 40$) for the turbulent flow.	75
4.8	Comparison of ground truth (GT) and POD-DL predicted vorticity fields for the turbulent flow at representative time steps from test cases T1 (a), T3 (b) and T4 (c). The index t_S refers to the snapshot number, not the physical time t	76

4.9	The top row shows the probability of achieving absolute vorticity prediction errors below thresholds of 0.2 (orange) and 0.5 (green) for the four test cases: T1 (a), T2 (b), T3 (c) and T4 (d), in the turbulent flow scenario. The bottom row depicts the temporal evolution of the spatially averaged streamwise velocity. These error thresholds are selected based on the typical vorticity range observed in the flow, as shown in Figure 4.8.	76
4.10	Turbulent kinetic energy (TKE) computed from the ground truth (left) and the adaptive framework predictions (middle) for test cases T1 (a), T2 (b), T3 (c) and T4 (d) in the turbulent flow scenario. The right panels display the absolute prediction error in TKE for each case.	77
4.11	Temporal evolution of the 0.25-th (bottom), 0.5-th (middle) and 0.75-th (top) vorticity percentiles for the ground truth (blue) and POD-DL predictions (orange) across test cases T1 (a), T2 (b), T3 (c) and T4 (d) in the turbulent flow scenario. Shaded regions denote the time intervals corresponding to snapshots used during the initial and retraining phases.	78
4.12	Percentage of energy contained, $E(k)$, when the truncation is done at the n , up to 20, most energetic modes in the isothermal jet.	80
4.13	Snapshots comparing ground truth (GT) and POD-DL predicted velocity fields for the isothermal subsonic jet at representative time steps from test cases T1 (a), T3 (b) and T4 (c). The index t_S denotes the snapshot number, not the physical time t	80
4.14	The top row shows the probability of achieving absolute velocity prediction errors below thresholds of 0.01 (orange) and 0.02 (green) for the four test cases: T1 (a), T2 (b), T3 (c) and T4 (d), in the isothermal jet scenario. The bottom row depicts the temporal evolution of the spatially averaged streamwise velocity. These error thresholds are selected based on the typical velocity range observed in the flow, as shown in Fig. 4.13.	81
4.15	Temporal evolution of the 0.25-th (bottom), 0.5-th (middle) and 0.75-th (top) velocity percentiles for the ground truth (blue) and POD-DL predictions (orange) across test cases T1 (a), T2 (b), T3 (c) and T4 (d) in the isothermal jet scenario. Shaded regions indicate the snapshot intervals used during both initial training and retraining phases.	82
5.1	Overview of the G-LED methodology: (1) high-resolution images (micro) are first downsampled via an encoder, (2) the resulting low-resolution (macro) dynamics are then forecasted using a transformer architecture and (3) the micro dynamics are subsequently reconstructed by applying a diffusion-based model.	89

5.2	Schematic illustration of the implementation of G-LED within the POD-DL methodology: As detailed in Chapter 2, (1) the original dataset, initially in tensor format, is reshaped into a matrix format, (2) POD is then applied to decompose this matrix into POD modes and coefficients, as described in eq. (2.10), enabling the retention of a selected number of POD modes and coefficients. (3) The number of retained POD coefficients is further reduced by the encoder, simplifying the dynamics to be predicted. (4) The reduced set of POD coefficients is then forecasted using the transformer model. (5) The diffusion model is employed to predict the POD coefficients that were discarded by the encoder. (6) Finally, the predicted snapshots are reconstructed by inverting the POD transformation using the original POD modes and the predicted POD coefficients.	91
5.3	Value of the R^2 metric, defined in eq. (5.12), for each prediction generated by G-LED.	92
5.4	Comparison of the GT (solid line) and G-LED predictions (dashed lines) for the root mean square (RMS) vorticity profile along the streamwise direction (a) and the mean vorticity alongside the streamwise direction (b).	92
5.5	Comparison between the ground truth (GT) and predicted vorticity snapshots for snapshots (a) 3210, (b) 3250 and (c) 3300 corresponding to the second steady-state in the flow.	93
5.6	Comparison of TKE computed from the ground truth (left) and from the predicted dynamics (middle) from G-LED model. The right column shows the absolute error between prediction and ground truth.	93
5.7	R^2 metric values, defined in eq. (5.12), for each prediction generated by G-LED when applied within the POD-DL methodology in the entire spatial domain.	94
5.8	Comparison of the GT (solid line) and G-LED, within the POD-DL methodology, predictions (dashed lines) for the root mean square (RMS) vorticity profile along the streamwise direction (a) and the mean vorticity alongside the streamwise direction (b).	95
5.9	Comparison between the ground truth (GT) and predicted vorticity snapshots for snapshots (a) 3210, (b) 3250 and (c) 3300 corresponding to the second steady-state in the flow. In this case predictions were obtained from G-LED implemented in the POD-DL methodology.	95
5.10	Comparison of TKE computed from the ground truth (left) and from the predicted dynamics (middle), corresponding to the application of the G-LED model within the POD-DL methodology. The right column shows the absolute error between prediction and ground truth.	96
5.11	Illustration of the spatial domain partitioned into two disjoint subdomains, within which POD and G-LED methodologies are independently applied. Following the work of Sengupta et al., 2025 a larger number of modes is retained close the cylinder than in regions further downstream.	96
5.12	R^2 metric values, defined in eq. (5.12), for each prediction generated by G-LED when applied within the POD-DL methodology in two different areas of the spatial domain.	97

5.13	Comparison of the GT (solid line) and G-LED, within the POD-DL methodology, predictions (dashed lines) for the root mean square (RMS) vorticity profile along the streamwise direction (a) and the mean vorticity alongside the streamwise direction (b). In this case the spatial domain is divided into two disjoint subdomains, as depicted in Fig. 5.11, with the G-LED methodology applied independently to each.	97
5.14	Comparison between the ground truth (GT) and predicted vorticity snapshots for snapshots (a) 3210, (b) 3250 and (c) 3300 corresponding to the second steady-state in the flow. In this case predictions were obtained from G-LED implemented in the POD-DL methodology and splitting the spatial domain in two disjoint areas as shown in Fig. 5.11.	98
5.15	Comparison of TKE computed from the ground truth (left) and from the predicted dynamics (middle), corresponding to the application of the G-LED model within the POD-DL methodology. In this case the spatial domain is divided into two disjoint subdomains, as depicted in Fig. 5.11, with the G-LED methodology applied independently to each. The right column shows the absolute error between prediction and ground truth.	98
A.1	Sliding window approach applied to the time series with a predefined stride. Each window comprises a sequence of preceding snapshots, serving as inputs to the forecasting model, and a subsequent snapshot, which constitutes the prediction target. In this illustrative example, m prior snapshots are utilized to forecast the immediate next state.	121
C.1	Snapshots of selected latent variables extracted from the encoder of the RAE, based on input data comprising the two velocity components of the flow past a circular cylinder at laminar regime (Le Clainche et al., 2018). For this dataset, the encoder receives as input a sequence of 10 prior snapshots with dimensions $[10 \times 100 \times 40 \times 3]$ and outputs a latent tensor of size $[10 \times 24 \times 18 \times 64]$. The figure displays channels 1, 2, 3, 4, 5, 10, 20, 35, 55, and 64 from top to bottom. The temporal evolution of each channel is shown from left to right over five consecutive time steps. All dimensions shown are in pixels, corresponding to the spatial resolution of the latent variables.	128
C.2	Snapshots of selected latent variables extracted from the encoder of the VAE, based on input data comprising the streamwise velocity component of the flow past a circular cylinder at laminar regime (Le Clainche et al., 2018). For this dataset, the encoder receives as input a sequence of 10 prior snapshots with dimensions $[10 \times 100 \times 40 \times 3]$ and outputs a latent tensor of size $[10 \times 24 \times 18 \times 64]$. The figure displays channels 1, 2, 3, 4, 5, 6, 9, 16, 25, and 32 from top to bottom. The temporal evolution of each channel is shown from left to right over five consecutive time steps. All dimensions shown are in pixels, corresponding to the spatial resolution of the latent variables.	129

C.3	Snapshots of selected latent variables extracted from the encoder of the VAE, based on input data comprising the normal velocity component of the flow past a circular cylinder at laminar regime (Le Clainche et al., 2018). For this dataset, the encoder receives as input a sequence of 10 prior snapshots with dimensions $[10 \times 100 \times 40 \times 3]$ and outputs a latent tensor of size $[10 \times 24 \times 18 \times 64]$. The figure displays channels 1, 2, 3, 4, 5, 6, 9, 16, 25, and 32 from top to bottom. The temporal evolution of each channel is shown from left to right over five consecutive time steps. All dimensions shown are in pixels, corresponding to the spatial resolution of the latent variables.	130
-----	---	-----

List of Tables

2.1	Distribution of training and testing samples used for the POD-DL model. . .	19
2.3	Normalizations applied to the POD-DL model, where each of them utilize different techniques. One of the normalization is applied directly on the dataset, before applying SVD, and the other on the POD coefficients, after SVD. . .	20
2.5	Number of POD modes retained based on the normalization technique choose for Norm ₁	23
3.1	Learning rates used for training each model. Note that the length of the input snapshot sequence varies by dataset for the RAE and the VAE models, whereas it remains fixed for the POD-DL model.	40
3.3	Number of snapshots used for both training and testing the models. Note the laminar flow is composed by three-dimensional snapshots, while the turbulent flow is composed by two-dimensional snapshots.	41
3.4	Training length measured in number of epochs, training time measured in minutes and prediction time of the time-ahead snapshots measured in seconds for each one of the three forecasting models and each dataset, respectively. Note the laminar flow is three-dimensional, and the turbulent flow is two-dimensional. Notably, the VAE model exhibited a markedly reduced training length for the turbulent dataset due to a rapid divergence early in training, after which convergence could not be restored regardless of the number of training epochs.	42
4.1	Architectural specifications of the POD-DL model. Here, k denotes the number of retained POD modes, m is the input window size (i.e., the number of past time steps used for forecasting) and B indicates the batch size used during training.	64
4.2	Key hyperparameters governing the behavior of the adaptive framework. A brief description of each hyperparameter is provided in the left column, with the corresponding symbol used throughout the text listed in the right column.	66
4.3	Hyperparameter settings used to evaluate the adaptive framework on the laminar flow around a circular cylinder. In the second test case, the number of predictions P is increased from $P = 100$ to $P = 200$ after the second training phase.	70
4.4	Hyperparameters used to test the adaptive framework on the turbulent flow.	74
4.5	Hyperparameters used to test the adaptive framework on the isothermal subsonic jet.	79

C.1	Architecture details in the POD-DL model. Unlike the other two models that receive snapshots as inputs, the POD-DL receive vectors representing the retained POD modes. In the case of the laminar flow is 20, as explained in Chapter 3. The term (6×) in the second and third layer, mean that 6 layers with the same configuration are trained in parallel. The architecture used in Chapter 2 is structurally identical, but employs a single layer instead of six, as it computes a single prediction.	126
C.2	Encoder architecture of the residual autoencoder (RAE) for input snapshots corresponding to the laminar flow dataset. Note that this architecture is resolution-dependent and may vary with the size of the input snapshots. Specifically, the number of Identity Blocks (Identity B.) and Convolutional Blocks (Conv B.) can be adjusted—added or removed—based on the spatial resolution of the data.	126
C.3	Same as Table C.2 for the ConvLSTM layer in the RAE.	126
C.4	Same as Table C.2 for the decoder in the RAE. As in the encoder, the number of transpose convolutional layers depends on the size of the input snapshots.	127
C.5	Encoder architecture of the variational autoencoder (VAE) for input snapshots corresponding to the laminar flow dataset. Note that this architecture is resolution-dependent and may vary with the size of the input snapshots. Specifically, the number of Identity Blocks (Identity B.) and Convolutional Blocks (Conv B.) can be adjusted—added or removed—based on the spatial resolution of the data.	127
C.6	Same as Table C.5 for the decoder in the VAE. As in the encoder, the number of transpose convolutional layers depends on the size of the input snapshots.	127

Abbreviations and acronyms

AI	Artificial Intelligence
CNN	Convolutional Neural Network
DL	Deep Learning
DMD	Dynamic Mode Decomposition
DNN	Deep Neural Network
GAN	Generative Adversarial Network
HODMD	Higher Order Dynamic Mode Decomposition
HPC	High Performance Computing
LES	Large Eddy Simulation
LLM	Large Language Model
LSTM	Long Short-Term Memory
ML	Machine Learning
MSE	Mean Squared Error
NLP	Natural Language Processing
POD	Proper Orthogonal Decomposition
RAE	Residual Autoencoder
RAN	Reynolds-Averaged Navier-Stokes
RL	Reinforcement Learning
RMSE	Relative Mean Squared Error
ROM	Reduced-Order Model
SSIM	Structural Similarity Index Measure
SVD	Singular Value Decomposition
UPM	Universidad Politécnica de Madrid
VAE	Variational Autoencoder

Chapter 1

Introduction

Fluid dynamics is the study of fluid motion, encompassing both liquids and gases. The study of liquid motion and its interactions with solid bodies is typically referred to as hydrodynamics, whereas the study of gases is known as aerodynamics. Research in fluid dynamics is traditionally categorized into three main approaches: theoretical, experimental, and computational, with the latter often relying on insights from the other two.

The fundamental mathematical framework of fluid dynamics is governed by the Navier-Stokes equations (Chorin and Marsden, 1979; Solomon, 2009), which are derived from three fundamental conservation principles: mass, momentum and energy. These equations are inherently nonlinear and exhibit significant spatial and temporal complexity, making analytical solutions feasible only in specific cases, such as the viscous Burgers equation, the Korteweg–de Vries (KdV) equation (Zabusky and Kruskal, 1965) or the Kuramoto-Sivashinsky equation.

For most practical problems, numerical methods are required to approximate solutions. In fluid dynamics, the study of such numerical methods fall under the field of computational fluid dynamics (CFD, Zawawi et al., 2018). These numerical approaches necessitate discretization in both time and space, leading to degrees of freedom on the order of thousands to millions. Consequently, the computational cost escalates as more accuracy is needed, which, in fluid dynamics, corresponds to the number of flow scales that must be resolved.

This number of flow scales increases approximately quadratically with the Reynolds number (Re), a dimensionless parameter defined as $Re = VD/\nu$, where V and D are the velocity and the characteristic length of the problem, respectively, and ν denotes the kinematic viscosity. Turbulent flows are typically characterized by high Reynolds numbers ($Re \gg 1$), except for viscoelastic fluids where turbulence appear at low Reynolds numbers, due to the coexistence of viscous and elastic effects (Foggi Rota et al., 2024).

Nevertheless, there is a significant interest in modeling complex flows, such as multi-scale, noisy, transitional and turbulent flows, due to their fundamental role in a wide range of natural and industrial processes, including combustion systems (Le Clainche et al., 2014a), power generation, aerodynamics (Marusic et al., 2003), marine propulsion (DeMont and Gosline, 1988), blood flow in tumors (Abadía-Heredia et al., 2021), heart flow dynamics (Lazpita et al., 2024) and the wakes of flying insects (Muijres et al., 2005), among others.

In many cases, turbulence induces undesirable effects, such as increased drag in vehicles, leading to higher fuel consumption and emissions, or structural vibrations in industrial devices due to fatigue loads. In other fields, like biomedical applications, turbulence has been associated with pathological conditions, including coronary artery diseases related to medical implants (Corrochano et al., 2021; Ferrari et al., 2006). Similarly, in urban environments, turbulence have been linked to pollutant dispersion within cities (Á. Martínez-Sánchez et al., 2023). Conversely, turbulence can also be beneficial, enhancing heat transfer and promoting fluid mixing, thereby improving efficiency in certain industrial systems (Le Clainche et al., 2014b).

The technique employed to numerically simulate all flow scales in these complex flows is known as a direct numerical simulation (DNS), which requires high computational costs (Brunton et al., 2020), giving rise to an entire field of research dedicated to their numerical simulation, known as high-performance computing (HPC). This field often requires specialized hardware and substantial computational time.

Nevertheless, recent breakthroughs in exascale HPC are redefining what is considered feasible in DNS. For instance, the work from Yeung et al., 2025 have shown that Graphics Processing Unit (GPU) Fourier pseudo-spectral solvers running on the world’s first exascale computer, Frontier, have already delivered isotropic-turbulence DNS with grid counts of order 10^{13} , sustaining over one exa-floating-point operation per second and reaching Taylor-scale Reynolds numbers close to 2500. Another study leveraging the parallel computing power of modern GPUs for HPC was presented by Kurz et al., 2025. They developed the GALÆXI solver, by combining high-order discontinuous-Galerkin methods and flux-reconstruction schemes. This solvers delivers high-order Discontinuous Galerkin Spectral Element Method on unstructured meshes, incorporates sub-cell finite-volume shock capturing, and shows near-ideal strong scaling up to 1024 GPUs, while halving the energy cost relative to a CPU baseline.

Efforts more focused on reducing the computational burden of turbulence simulation have led to the development of coarse-graining approaches, such as large eddy simulations (LES), which filter out smaller turbulent scales (Moin, 2002; Piomelli, 1999), or Reynolds-averaged Navier–Stokes (RANS) methods, which employ time-averaging of the Navier-Stokes equations to model turbulence effects (Alfonsi, 2009).

Complementing high-order schemes, adaptive mesh refinement frameworks (AMR) allow the grid resolution to be concentrated only where physics demands it. Recently, Y. Wang et al., 2025 introduced a heterogeneous GPU-accelerated AMR framework based on AMReX, including subcycling-in-time support and a specialized refluxing algorithm. This work shown 6-to-7 $\times$ speed-ups, compared to the CPU implementation, over uniform grids while maintaining second-order accuracy and excellent multi-GPU scaling.

However, the computational cost of high- and full-order simulations, such as DNS or even LES or RANS, remains still prohibitive for many cases due to the vast number of degrees of freedom required to resolve all flow features. This challenge is particularly pronounced in scenarios where traditional methods necessitate repeated model evaluations over a broad range of parameters, like in flow control problems (Benner et al., 2015; Rowley and Dawson, 2017).

Therefore, reducing these high computational costs while preserving the required precision in the modeled dynamics remains a key challenge.

In this context, it has been observed that computing all flow scales is often unnecessary in many practical scenarios (Jiménez, 2018). This insight has led to the development of reduced-order models (ROMs), which enable the solution of the governing equations only for the scales of interest, significantly reducing computational complexity. These ROMs are designed to serve as surrogate models of the full-order governing equations. Consequently, a fundamental objective in such ROM approaches is to assess their ability to accurately reproduce the underlying flow dynamics.

These ROMs can be classified into two types: intrusive and non-intrusive. Both approaches project the solution of the governing equations onto a lower-dimensional representation. Intrusive ROMs, also known as projection-based ROMs, directly operate on the governing equations by projecting them onto a reduced basis, as seen in Galerkin methods (Luchtenburg et al., 2009a; Noack et al., 2011a; Quarteroni et al., 2016). In contrast, non-intrusive ROMs, or data-driven ROMs, rely purely on data-driven techniques to extract low-dimensional representations from the data itself, without explicitly modifying the governing equations (Le Clainche and Vega, 2017; Lumley, 1967; Schmid, 2010).

The basic philosophy of projection-based ROM approaches is to reduce the high dimensionality of the governing equations by projecting them onto a transformed space defined by an orthonormal basis. Among these techniques, Proper Orthogonal Decomposition (POD, Lumley, 1967) has emerged as a widely used method for constructing this orthonormal basis in dynamical systems (Antoulas et al., 2001; Benner et al., 2015; Berkooz et al., 1993; Holmes et al., 1996; Rowley and Dawson, 2017). It identifies the most dominant flow features by capturing the most energetic modes. POD was first applied to a fluid dynamic problem by Lumley, 1967, who used this technique to find coherent structures within the dynamics of a turbulent flow (Liu, 1988).

The evolution equations for the low-dimensional system are then derived using the Galerkin method. For many flows, the POD-Galerkin approach offers an efficient and accurate framework for constructing ROMs (Luchtenburg et al., 2009b; Noack et al., 2011b; Rahman et al., 2018; Rowley et al., 2004). This low-dimensional representation of the dynamics is obtained by truncating the number of modes based on their energy content. However, since truncated modes account for unresolved flow dynamics, certain effects are not explicitly modeled. To address this limitation, various closure models have been proposed to approximate the influence of the discarded modes in the overall dynamics (San and Iliescu, 2013; Z. Wang et al., 2012).

Nevertheless, modeling the effects neglected due to mode truncation still remain one of the primary challenges in Galerkin methods. For instance, as recently discussed by Reiss et al., 2018, transport-dominated phenomena cannot be accurately captured by a limited number of dominant modes, necessitating a larger mode set to better recover the underlying structures of the system. However, increasing the number of modes also raises computational costs, potentially diminishing the efficiency of the ROM.

These limitations primarily stem from the linear nature of mode computation. To address

this, nonlinear data-driven models have recently been proposed, enabling the decomposition of high-dimensional dynamics into low-dimensional representations. For instance, Ehlert et al., 2019 introduced a manifold representation of the transient oscillatory cylinder wake using a locally linear embedding approach as an encoder. Their results demonstrated that this representation outperformed a 50-dimensional POD expansion derived from the same data.

Another fundamental challenge in Galerkin-based methods is the assumption that modes are globally coupled, which can lead to an inherent mismatch between the Navier-Stokes equations and Galerkin-based ROMs. Additionally, the frequency range present in high-dimensional Navier-Stokes solutions is often not fully captured in the low-dimensional deterministic system (Bourgeois et al., 2013), further limiting the accuracy of the ROM.

For example, Rempfer, 2000 investigated the implications of projecting the Navier-Stokes equations onto a lower-dimensional basis and demonstrated how constraints introduced by such projections, along with improper treatment of boundary conditions, can compromise the accuracy and validity of the resulting ROM.

Due to all these limitations, there is a recent interest in developing fully non-intrusive ROMs that do not require access to the full-order governing equations, in order to establish surrogate models. For instance, the POD can also be considered a non-intrusive ROM, as its mode computation can be performed purely in a data-driven manner using techniques such as singular value decomposition (SVD, Sirovich, 1987) or spatio-temporal biorthogonal decomposition (Aubry et al., 1991). Additionally, non-intrusive variants of POD include spectral POD (Sieber et al., 2016) and multiscale POD (Mendez et al., 2019).

Other widely used purely data-driven ROMs include dynamic mode decomposition (DMD, Schmid, 2010) and higher-order dynamic mode decomposition (HODMD, Le Clainche and Vega, 2017). These models are categorized in the literature as modal decomposition techniques, a comprehensive review of these non-intrusive ROMs applied to fluid dynamics was presented by Begiashvili et al., 2023.

The advent of deep learning (DL, Goodfellow et al., 2016) algorithms has significantly expanded the potential for developing non-intrusive ROMs for flow dynamics prediction (Brunton et al., 2020; Kutz, 2017). Several studies have explored the application of DL techniques to improve ROM performance and address challenges associated with mode truncation and model accuracy.

For instance, San and Maulik, 2018a employed a DL model to dynamically compute an eddy-viscosity closure, incorporating the effects of truncated modes in Galerkin-based ROMs. Similarly, Heimbach et al., 2025 used reinforcement learning (RL, Sutton and Barto, 2018) to develop closures models for coarse-grained PDEs, and Bae and Koumoutsakos, 2022 also applied RL for the discovery of wall models for LES.

In another study, San and Maulik, 2018b investigated the effectiveness of DL models in capturing the influence of truncated modes in a nonlinear advection-diffusion system. To construct a fully data-driven ROM for fluid dynamics, Xie et al., 2019 proposed linear multi-step networks based on implicit Adams-Moulton schemes. Omata and Shirayama, 2019 developed a convolutional autoencoder to create a low-dimensional representation for

spatio-temporal data analysis of unsteady flow fields.

A machine learning (ML) approach for identifying nonlinear dynamical systems by integrating multi-step time-stepping schemes with DL architectures was introduced by Raissi et al., 2018 to extract governing mechanisms directly from data. Kani and Elsheikh, 2017 developed a deep residual recurrent neural network as an efficient model reduction technique for nonlinear dynamical systems, while Yeo and Melnyk, 2019 combined a long short-term memory (LSTM) architecture (Hochreiter and Schmidhuber, 1997) with numerical discretization to simulate stochastic processes with underlying nonlinear dynamics.

Additionally, Qin et al., 2019 proposed several data-driven frameworks using DL models to approximate unknown governing equations, with a distinctive focus on formulations based on the integral form of the equations. Buaria and Sreenivasan, 2023 demonstrated a DL-based approach for predicting small-scale structures in turbulent flows at high Reynolds numbers by training on data from lower and moderate Reynolds number flows. Finally, Cheng et al., 2020 utilized a generative adversarial network (GAN, Goodfellow et al., 2014) to predict water depth fields for various coastline configurations, relying solely on high-fidelity simulation data for training.

In these cases, the governing equations did not require modification to predict new dynamics, instead, the learned data-driven models effectively captured the underlying physics and extrapolated the flow behavior. These models fall under the category of deep learning-based reduced-order models (DL-ROMs), as they leverage deep learning techniques to construct ROMs that approximate the dynamics described by full-order governing equations. As surrogate models, DL-ROMs offer substantial reductions in computational cost while maintaining the ability to accurately predict complex flow dynamics.

In other scenarios, DL-ROM methodologies can be employed to forecast flow dynamics based on historical data, significantly reducing the computational cost associated with obtaining the required solutions. This PhD thesis primarily focuses on the study of such models applied to fluid dynamics. Accordingly, a brief review of the relevant literature on these methodologies is presented below.

Time series forecasting is typically categorized into two distinct subfields (Januschowski et al., 2020): point forecasting and probabilistic forecasting. In this context, point forecasting models represent the prediction process as a deterministic function that approximates future states. Specifically, these models take a past sequence of samples as input to produce the time-ahead predictions.

In contrast, probabilistic forecasting (Gneiting and Katzfuss, 2014) seeks to estimate the probability distribution of future states rather than providing a single deterministic outcome. While these models also receive a past sequence of samples as input, they use this information to compute key parameters of a probability distribution, from which the time-ahead predictions are sampled.

Time series forecasting models in fluid dynamics has been extensively explored in the literature, leading to the development of various deep learning models and methodologies. In the context of probabilistic forecasting, several models have been proposed for predicting flow dynamics.

For instance, Temirchev et al., 2020 combined a variational autoencoder (VAE, D. P. Kingma and Welling, 2013) with recurrent neural networks to forecast the dynamics of multiphase fluid flows in oil reservoirs. In this approach, the probabilistic nature of the model is introduced through the VAE, which encodes the latent space as a probability distribution.

A similar methodology was employed by Wu et al., 2022, who integrated a VAE with a convolutional neural network-based forecasting model to predict the evolution of flows around a cylinder at different Reynolds numbers.

Following in the probabilistic framework, a recent class of models known as diffusion models has gained popularity due to their ability to approximate complex distributions, they demonstrated the capability to surpass state-of-the-art models at the time, such as VAEs and GANs (Dhariwal and Nichol, 2021). Due to this ability diffusion models have been widely applied to time series forecasting (Lin et al., 2024a) and fluid dynamics (Gao, Kaltenbach, and Koumoutsakos, 2024; Jacobsen et al., 2024; Kohl et al., 2024; Lippe et al., 2023a; Rühling Cachay et al., 2023; Shysheya et al., 2024).

Within the point forecasting framework, Pathak et al., 2018 employed an echo-state network based on recurrent neural networks to predict the state evolution of chaotic systems. Similarly, Vlachas et al., 2018 demonstrated the effectiveness of LSTM networks for forecasting chaotic dynamics.

Other approaches have integrated convolutional autoencoders with LSTM networks to capture both spatial and temporal features of spatio-temporal flow dynamics, as demonstrated by Han et al., 2019 and Hasegawa et al., 2020. Nakamura et al., 2021 further extended this methodology to three-dimensional flow fields, enhancing the model’s capability to handle complex spatial patterns.

The widespread adoption of DL autoencoders in fluid dynamics is primarily driven by the spatio-temporal nature of the data. These models project high-dimensional flow dynamics onto a low-dimensional latent space, effectively capturing underlying structures and patterns while enabling the decomposition of the flow into meaningful components (Omata and Shirayama, 2019). In this context, Hasegawa et al., 2020 studied the role of latent spaces in forecasting high-dimensional dynamics.

This projection to a lower-dimensional space is analogous to traditional modal decomposition techniques like POD or DMD. Notably, for linear autoencoders, it has been observed that the computed latent variables closely resemble the modes in POD (Milano and Koumoutsakos, 2002). However, nonlinear autoencoders offer superior compression capabilities compared to modal decomposition techniques, although often at the expense of interpretability regarding the underlying physical phenomena (Eivazi et al., 2022; Fukami, Fukagata, and Taira, 2020; Fukami, Nakamura, and Fukagata, 2020; Muñoz et al., 2023; Murata et al., 2020).

Moreover, unlike modal decomposition techniques, DL autoencoders require a training phase to capture the underlying flow structures, which may increase the amount of data needed.

To address these challenges, hybrid models have been explored, where DL autoencoders are replaced by modal decomposition techniques to enhance interpretability and reduce data requirements. For instance, Ali et al., 2021 proposed a hybrid model that integrates dynamic

mode decomposition with control (DMDc) and DL architectures.

Similarly, POD can be employed to reduce dimensionality, with forecasting performed on the POD coefficients to predict the temporal evolution of the corresponding modes. Several studies have investigated such POD-DL hybrid models, combining POD with DL architectures (Abadía-Heredia et al., 2022; Abadía-Heredia, Corrochano, et al., 2024; Besabe et al., 2025; Corrochano et al., 2023; Mohan and Gaitonde, 2018; Parish and Carlberg, 2020; Pawar et al., 2019; Regazzoni et al., 2019; Xiao et al., 2019; Xu et al., 2022). Furthermore, a probabilistic forecasting version of these hybrid POD-DL models has been proposed by Maulik et al., 2020.

This thesis has focused on the study of these hybrid models (POD-DL), which combine POD with DL architectures, for the temporal prediction of fluid dynamics. Therefore, the thesis is organized as follows: Chapter 2 introduces the POD-DL framework and examines the influence of various data scaling strategies on its performance. Chapter 3 compares the hybrid POD-DL model against two purely DL autoencoders, highlighting the enhanced predictive capabilities of the hybrid approach. Building on these findings, Chapter 4 introduces an adaptive retraining pipeline that intermittently incorporates high-fidelity data to update the POD-DL model, thereby enhancing its predictive capability for long-term forecasts while significantly reducing the computational cost relative to traditional flow solvers. Chapter 5 explores the application of generative diffusion models for capturing and forecasting the small-scale structures of highly turbulent flows. Finally, Chapter 6 synthesizes the principal conclusions derived from this research.

1.1 Research contributions and novelty

In recent years, the field of artificial intelligence (AI) has witnessed remarkable advancements, particularly in natural language processing (NLP, Khurana et al., 2023), where large language models (LLMs), such as GPT-2 (Radford et al., 2019), GPT-3 (Brown et al., 2020), DeepSeek-R1 (DeepSeek-AI et al., 2025) or Claude (Anthropic, 2024), have been developed as highly capable conversational agents. Additionally, significant progress has been achieved in computer vision and other subfields. In aerospace engineering, the use of RL and LLMs have shown substantial promise in the autonomous control of unmanned aerial vehicles (UAVs, Giral et al., 2025) and spacecraft (Carrasco et al., 2025; Rodriguez-Fernandez et al., 2024), as well as in trajectory generation (Briden et al., 2025).

Despite these advances, a common and critical challenge across many of these domains is the dependency on large and diverse datasets for training. In NLP, for instance, pretraining typically relies on vast datasets created from internet sources (Penedo et al., 2024). Similarly, in aerospace applications, training data are often generated using real-time numerical simulations that emulate vehicle dynamics in varied environments (Carrasco et al., 2025).

However, in fluid dynamics data acquisition is particularly resource-intensive. High-fidelity datasets are usually obtained through computationally expensive numerical simulations or costly experiments. For example, state-of-the-art AI models for turbulence prediction may require a number of training samples up to the order of 100,000 (Lippe et al., 2023b; Shysheya et al., 2024) to achieve accurate predictions.

The principal challenge addressed in this thesis is the development of data-efficient learning methodologies in fluid dynamics, where datasets are limited in the number of available samples. Specifically, the datasets used here range from a minimum of 499 to a maximum of 5000 high-dimensional samples. To address this, the core contribution of this thesis lies in the design of a hybrid model methodology (Chapter 2) that reduces the spatial dimensionality, thereby enabling the training of simpler models with significantly fewer data samples, as shown in Chapter 3.

Furthermore, recognizing the limitations of such hybrid methodology, this thesis explores adaptive learning strategies (Chapter 4) and generative AI hybrid models (Chapter 5) in order to further mitigate the challenges posed by high-dimensional inputs and a small number of training samples. Therefore, the main innovation of this thesis is the proposal and study of AI-based methodologies capable to generate accurate predictions of future states in laminar and, specifically, turbulent flow regimes, while operating with strict data limitations.

1.2 Publications published during the thesis, presentations at conferences and patent

The publications made during the thesis, as well as the presentations given at conferences and the patent granted from some of the work done in the thesis can be found listed below.

1.2.1 Publications

1. Rodrigo Abadía-Heredia, Manuel Lopez-Martin and Soledad Le Clainche. (2025). An Adaptive Framework for Autoregressive Forecasting in CFD Using Hybrid Modal Decomposition and Deep Learning. arXiv. <https://arxiv.org/abs/2505.01531>.
2. Guillermo Barragán, Ashton Hetherington, Rodrigo Abadía-Heredia, Jesús Garicano-Mena and Soledad Le Clainche. (2025). HOSVD-SR: A Physics-Based Deep Learning Framework for Super-Resolution in Fluid Dynamics. arXiv, <https://arxiv.org/abs/2504.17994>.
3. Arindam Sengupta, Rodrigo Abadía-Heredia, Ashton Hetherington, José Miguel Pérez and Soledad Le Clainche. (2025). Hybrid machine learning models based on physical patterns to accelerate CFD simulations: a short guide on autoregressive models. arXiv, <https://arxiv.org/abs/2504.06774>.
4. Rodrigo Abadía-Heredia, Adrian Corrochano, Manuel Lopez-Martin and Soledad Le Clainche. (2025). Generalization capabilities and robustness of hybrid models grounded in physics compared to purely deep learning models. *Physics of Fluids*, 37(3), p.035149, <https://doi.org/10.1063/5.0253876>.
5. Rodrigo Abadía-Heredia, Manuel Lopez-Martín y Soledad Clainche. (2024). Evaluating Forecast Divergence in Fluid Dynamics: A Metric for Autoregressive Models in the Absence of True Solution Data. 9th European Congress on Computational Methods in Applied Sciences and Engineering (ECCOMAS), <https://www.scipedia.com/public/>

Abadia-Heredia_et_al_2024a

6. Ashton Hetherington, Adrian Corrochano, Rodrigo Abadía-Heredia, Eneko Lazpita, Eva Muñoz, Paula Díaz, Egoitz Maiora, Manuel López-Martín and Soledad Le Clainche. (2024). ModelFLOWS-app: data-driven post-processing and reduced order modelling tools. *Computer Physics Communications*, 301, p.109217, <https://doi.org/10.1016/j.cpc.2024.109217>.
7. León Mata, Rodrigo Abadía-Heredia, Manuel Lopez-Martin, José M. Pérez and Soledad Le Clainche. (2023). Forecasting through deep learning and modal decomposition in two-phase concentric jets. *Expert Systems with Applications*, 232, p.120817, <https://doi.org/10.1016/j.eswa.2023.120817>.
8. Rodrigo Abadia-Heredia, Marco Cialesi-Esposito, Luca Brandt and Soledad Le Clainche. (2022). Shallow Neural Networks and Turbulent Flows. *International Symposium on Unmanned Systems and The Defense Industry (ISUDEF)*, Sustainable Aviation, Springer, https://doi.org/10.1007/978-3-031-37160-8_15.
9. León Mata, Rodrigo Abadía-Heredia, Jose Miguel Pérez and Soledad Le Clainche. (2022). Neural Networks to Speed Up Multiphase Flow Numerical Simulations. *International Symposium on Unmanned Systems and The Defense Industry (ISUDEF)*, Sustainable Aviation, Springer, https://doi.org/10.1007/978-3-031-37160-8_36.
10. Rodrigo Abadía-Heredia, Manuel López-Martín, Belen Carro, J. Ignacio Arribas, J. Miguel Pérez y S. Le Clainche. (2022). A predictive hybrid reduced order model based on proper orthogonal decomposition combined with deep learning architectures. *Expert Systems with Applications*, 187, p.115910, <https://doi.org/10.1016/j.eswa.2021.115910>.
11. Rodrigo Abadía-Heredia, Ana Pariente, José M. Pérez and Soledad Le Clainche 2021. Tortuosity in tumours: The need of combining multi-phase flows with machine learning tools. *Results in Engineering*, 11, p.100234, <https://doi.org/10.1016/j.rineng.2021.100234>.

1.2.2 Presentations at conferences

1. Evaluating Forecast Divergence in Fluid Dynamics: A Metric for Autoregressive Models in the Absence of True Solution Data. (2024). 9th European Congress on Computational Methods in Applied Sciences and Engineering (ECCOMAS), Lisbon, Portugal.
2. Learning extrapolation in the reconstruction and forecasting of a turbulent velocity flow field using Autoencoders and Singular Value Decomposition. (2023). 2nd Spanish Fluid Mechanics Conference (SFMC), Barcelona, Spain.
3. Reconstruction and Forecasting of a turbulent velocity flow field through Autoencoders and Singular Value Decomposition. (2023). Math 2 Product (M2P). Taormina, Sicily, Italy.
4. A two-phase mixing layer between parallel gas and liquid streams: Flow structures analysis and deep learning. (2022). 1st Spanish Fluid Mechanics Conference (SFMC),

Cádiz, Spain.

5. Shallow Neural Networks and Turbulent Flows. (2022). International Symposium on Unmanned Systems and The Defense Industry (ISUDEF). Online.
6. A predictive hybrid model based on proper orthogonal decomposition combined with deep learning. (2022). Workshop Junior Interdisciplinar. Madrid, Spain.
7. Data-driven reduced-order models in fluid dynamics. (2021). XIX Jacques-Louis Lions Spanish-French School on Numerical Simulation in Physics and Engineering, Madrid, Spain.

1.2.3 Patent

1. Ashton Ian Hetherington, Adrian Corrochano Calcerrada, Santos Rodrigo Abadia Heredia, Eneko Lazpita Suinaga and Soledad Le Clainche Martinez. (2025). Método implementado por ordenador para el procesamiento de imágenes. ES2980275A1.

Chapter 2

The POD-DL: A hybrid methodology combining modal decomposition and deep learning for flow dynamics forecasting.

As detailed in the introduction, the POD-DL model introduced in this chapter constitutes one of the core contribution of this thesis. This hybrid methodology combines proper orthogonal decomposition (POD) with DL architectures to develop a non-intrusive reduced-order model (ROM) for forecasting flow dynamics. Inspired by Galerkin methods, the model employs POD to project the high-dimensional flow dynamics onto a lower-dimensional manifold. Deep learning architectures are then leveraged within this reduced space to predict the temporal evolution of the projected dynamics. Finally, the original high-dimensional representation is reconstructed by inverting the POD, enabling the computation of full-order flow predictions. As later detailed in Chapter 3, this hybrid model offers advantages to other purely DL models commonly used in fluid dynamics to predict flow behaviors.

This chapter is structured as follows: Section 2.1 discusses the motivation for the hybrid methodology, while Section 2.2 provides a detailed description of the POD-DL model. Section 2.3 presents the results obtained by applying this approach to a dataset describing a laminar flow past a circular cylinder, with different normalization techniques implemented to assess the model's sensitivity to normalization. Finally, Section 2.4 offers a discussion of the findings.

2.1 Projection-based ROM

To introduce the motivation behind the POD-DL model, consider the spatial and temporal domains $\mathcal{P}_x \subset \mathbb{R}^3$ (or $\mathbb{R}^2$ for two-dimensional cases) and $\mathcal{P}_t \subset \mathbb{R}$, respectively, which define the compact set $\mathcal{P} = (\mathcal{P}_x, \mathcal{P}_t)$ where the system of PDEs is solved.

Let $\mathcal{H} = \mathcal{H}(\mathcal{P})$ be a Hilbert space defined over $\mathcal{P}$, representing the space of all possible solutions to the system of PDEs on this domain. That is, $\mathcal{H}$ consists of functions $\mathbf{v}$ such that

$\mathbf{v} : \mathcal{P}_x \times \mathcal{P}_t \rightarrow \mathbb{R}^n$, where $n \in \mathbb{N}$.

The system of nonlinear PDEs can be then denoted as a mapping $\Phi : \mathcal{H} \times \mathcal{P} \rightarrow \mathbb{R}^n$, where the problem of interest reads: given $p = (x, t) \in \mathcal{P} = (\mathcal{P}_x, \mathcal{P}_t)$, find a $\mathbf{v}(p) \in \mathcal{H}(\mathcal{P})$ such that,

$$\Phi(\mathbf{v}(p), p) = (0, \dots, 0). \quad (2.1)$$

In practical applications, rather than seeking an exact solution $\mathbf{v} \in \mathcal{H}$ that satisfies (2.1) exactly, the objective is to find an approximation that minimizes the residual error, making (2.1) as close to zero as possible.

Discretizing the system of equations Φ enables the application of numerical methods, such as finite element or finite volume methods (Tadmor, 2012), to approximate the solution. In fluid dynamics, this approach is referred to as direct numerical simulation (DNS) (Doran, 2013). However, DNS resolves all spatio-temporal scales permitted by the spatial grid, including fine-scale structures that may not be relevant to the specific problem under study. This leads to a significant increase in computational cost, as resources are spent computing information that is not necessarily required.

In this context, Galerkin methods provide a framework for projecting the discretized equations onto an orthonormal basis. This is achieved by expressing the solution $\mathbf{v}(x, t)$ as a linear combination of orthonormal modes, obtained using POD, as follows,

$$\mathbf{v}(x, t) = \sum_{i=1}^K a_i(t) \mathbf{U}_i(x). \quad (2.2)$$

Here, the set $\{\mathbf{U}_1, \dots, \mathbf{U}_K\}$ represents the orthonormal basis, or POD modes, while a_i denotes the mode amplitudes, which quantify the energy contribution of each POD mode to the overall flow dynamics (Aubry et al., 1991). These amplitudes allow for the classification of POD modes based on the amount of information they capture about the system's dynamics. This technique has been widely used to analyze the structural composition of flow fields, facilitating the identification of coherent structures within turbulence (Holmes et al., 1996). The first application of POD to a turbulent flow was introduced by Lumley in 1967 (Lumley, 1967).

By truncating the number of modes and retaining only a reduced subset, $k < K$, based on their energy contribution, a new approximation $\mathbf{v}^k(x, t)$ is defined, capturing the essential dynamics contained within the first k modes. Substituting $\mathbf{v}^k(x, t)$ into the discretized version of (2.1) yields a reduced system of equations, denoted as Φ_{GP} , with a significantly lower number of degrees of freedom. The complexity of this reduced system is directly proportional to the number of POD modes retained in the truncation.

This reduced-order system of equations depends on the mode amplitudes $\mathbf{a}(t) = \mathbf{a}^t = [a_1(t), \dots, a_N(t)]^\top$ rather than the full solution $\mathbf{v}^k(x, t)$. Defining $\mathcal{A}$ as the space containing all possible mode amplitudes, the reduced system can be expressed as a mapping $\Phi_{GP} : \mathcal{A} \times \mathcal{P}_t \rightarrow \mathbb{R}^n$. The problem is now reformulated as follows: given $t \in \mathcal{P}_t$, find $\mathbf{a}(t) \in \mathcal{A}(\mathcal{P}_t)$ such that,

$$\Phi_{GP}(\mathbf{a}(t), t) = (0, \cdot^{\cdot^{\cdot^{\cdot^{\cdot}}^n}, 0). \quad (2.3)$$

Once (2.3) is solved, the solution $\mathbf{v}^k(x, t)$ is reconstructed using (2.2). Thus, Galerkin methods focus on computing the amplitude dynamics $\mathbf{a}(t)$, which describe the temporal evolution of the POD modes.

These Galerkin methods are classified as intrusive since they require modifications to the full-order equations when projecting them onto the POD modes to construct the reduced-order system. In contrast, non-intrusive approaches do not alter the governing equations, as they rely entirely on data-driven techniques. As discussed in the Introduction (Chapter 1), purely data-driven predictive models typically operate on datasets derived from numerical simulations, which solve the full-order equations, or experimental measurements.

These models often employ techniques that decompose the spatiotemporal information into its distinct patterns, facilitating temporal predictions. In DL, this is commonly achieved through autoencoders, which leverage convolutional neural networks (CNNs). This approach is further explored in Chapter 3.

However, the decomposition performed by autoencoders lacks interpretability concerning the underlying physical dynamics (Eivazi et al., 2022; Fukami, Fukagata, and Taira, 2020; Fukami, Nakamura, and Fukagata, 2020; Muñoz et al., 2023; Murata et al., 2020). An alternative approach is the use of POD, which enables the decomposition of flow dynamics into distinct POD modes and their corresponding amplitudes, as expressed in (2.2). The temporal evolution of the system can then be predicted by modeling the time-dependent behavior of these amplitudes, which encapsulate the temporal dynamics of the modes.

This modeling can be achieved using a data-driven approach, such as a DL model, which learns the temporal correlations within the POD amplitudes and extrapolates their evolution. The reconstruction of the full-order dynamics is then performed via the matrix multiplication defined in (2.2). This hybrid, non-intrusive methodology is identified as POD-DL, since it combines POD and DL models. It is similar to Galerkin methods in the way that both of them perform POD and model the temporal evolution of the POD amplitudes, in order to compute the time-ahead evolution of POD modes.

A detailed explanation of the POD-DL framework, including the computation of POD modes and the DL models employed, is provided in the following section.

2.2 The POD-DL methodology

As detailed in the previous section, the POD-DL method is a hybrid, non-intrusive ROM that integrates POD with DL architectures to predict the temporal evolution of flow dynamics, and it can be visually depicted in Fig. 2.1. While it shares similarities with Galerkin methods in that both operate on POD amplitudes, the POD-DL approach differs fundamentally by relying entirely on datasets rather than directly incorporating the governing equations. Consequently, the POD modes and amplitudes must be computed in a fully data-driven manner to ensure seamless integration with DL models.

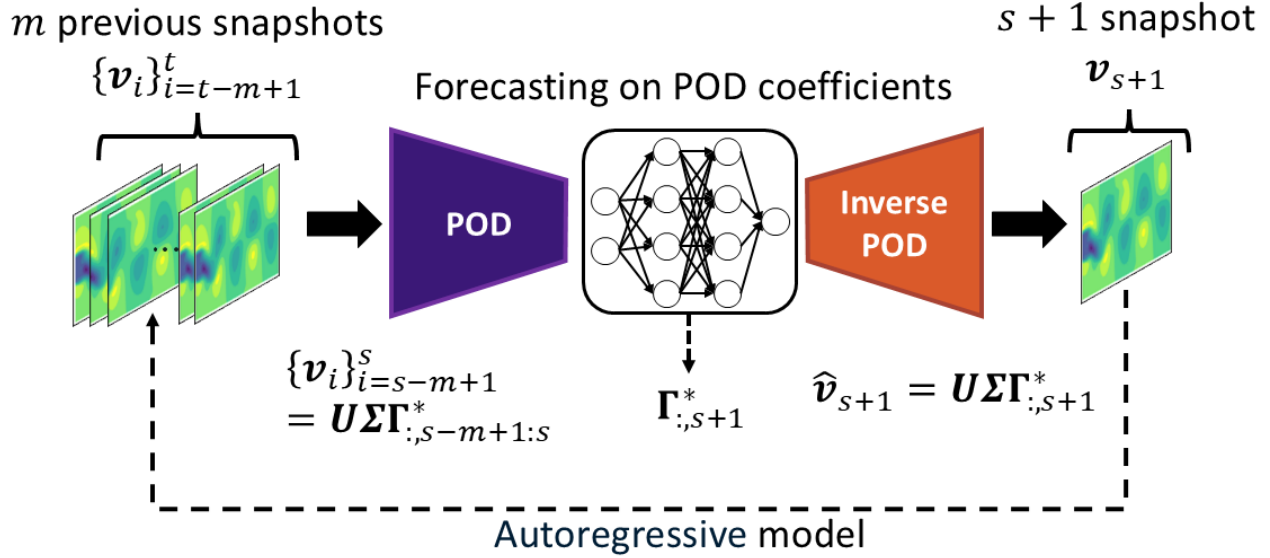


Figure 2.1: The POD-DL framework.

2.2.1 Implementation of POD

In this context, the POD modes are computed using a full data-driven technique known as singular value decomposition (SVD), commonly referred to as the method of snapshots, introduced by Sirovich, 1987.

This technique decomposes a dataset $\mathring{\mathbf{D}}$, which is composed of a series of snapshots representing the flow dynamics, into its respective POD modes and amplitudes. The snapshots forming the dataset can correspond to velocity fields, pressure fields, or other relevant flow variables. In numerical applications like CFD, the spatial resolution of the snapshots is usually determined by the spatial grid used for discretizing the domain.

Let the spatial domain be discretized into $N = \#\mathcal{P}_x$ points, where $\mathcal{P}_x \subset \mathbb{R}^3$ (or $\mathbb{R}^2$) and let T_1 represent the number of available snapshots. Here, T_1 is less than the total temporal horizon $T = \#\mathcal{P}_t$, over which we aim to solve (2.1). Under these conditions, the dataset $\mathring{\mathbf{D}}$ is structured as a snapshot tensor, defined as follows,

$$\mathring{\mathbf{D}} = [\mathbf{v}(\mathcal{P}_x, t_1), \dots, \mathbf{v}(\mathcal{P}_x, t_{T_1})], \quad (2.4)$$

$$\dim(\mathring{\mathbf{D}}) = [C \times N_x \times N_y \times N_z \times T_1]. \quad (2.5)$$

Here, $\dim$ denotes the dimensionality of $\mathring{\mathbf{D}}$, while C represents the number of components, such as velocity components (streamwise, wall-normal and spanwise), different Reynolds numbers, or other flow parameters. The variables N_x , N_y and N_z correspond to the number of discretization points along the x -, y - and z -axes, respectively, such that, $N = N_x * N_y * N_z$.

It is important to emphasize that this dataset consists of solutions to the system of PDEs given by (2.1), where typically $T_1 \ll T$. As a result, generating the required snapshots for

SVD necessitates solving (2.1) over a minimal time horizon T_1 , ideally ensuring that $T_1 \ll T$ to reduce the computational cost.

Once the dataset $\tilde{\mathbf{D}}$ is obtained, it has to be reshaped into a matrix representation, $\mathbf{D}$, with dimensionality $\dim(\mathbf{D}) = [C * N_x * N_y * N_z \times T_1]$. This transformation is necessary because SVD operates exclusively on matrix-structured data. Another common practice is to subtract the temporal mean, $\bar{\mathbf{D}}$, from $\mathbf{D}$, as follows,

$$\bar{\mathbf{D}} = \frac{1}{T_1} \sum_{i=1}^{T_1} \mathbf{D}_{:,i}, \quad (2.6)$$

$$\dim(\bar{\mathbf{D}}) = [C * N_x * N_y * N_z \times 1]. \quad (2.7)$$

Here the subscript ":", i denotes taking all elements in the i -th column of matrix $\mathbf{D}$. The mean matrix $\bar{\mathbf{D}}$ is also commonly known in fluid dynamics as the mean flow, and it has a single element in the temporal dimension as the mean has been computed in this dimension. Therefore, the subtraction of the mean flow from the original dataset corresponds to the fluctuations within the flow dynamics, and is defined as follows,

$$\tilde{\mathbf{D}} = \mathbf{D} - \bar{\mathbf{D}}, \quad (2.8)$$

$$\dim(\tilde{\mathbf{D}}) = [C * N_x * N_y * N_z \times T_1]. \quad (2.9)$$

The purpose of computing the fluctuations within the dynamics is to ensure that SVD is applied exclusively to the time-evolving structures. Without this step, the most energetic mode (i.e., the first POD mode) would correspond to the mean flow. Once the fluctuations are extracted, SVD can be performed on $\tilde{\mathbf{D}}$ to obtain the following decomposition,

$$\tilde{\mathbf{D}} = \mathbf{U} \mathbf{\Sigma} \mathbf{\Gamma}^*. \quad (2.10)$$

Here, $\mathbf{U}$ is a unitary matrix whose columns correspond to the POD modes in (2.2), $\mathbf{\Sigma}$ is a diagonal matrix containing the singular values. The matrix $\mathbf{\Gamma}$ is also unitary, with its columns representing the POD coefficients, and $\mathbf{\Gamma}^*$ denotes its conjugate transpose. Since $\mathbf{\Sigma}$ is diagonal, (2.10) can be expressed as a sum of rank-one matrices, as follows,

$$\tilde{\mathbf{D}} = \sum_{i=1}^K \sigma_i \mathbf{u}_i \gamma_i^*, \quad (2.11)$$

where $\sigma_i \in \mathbb{R}^+$ is nonnegative and represents the i -th singular value (i.e., the i -th element in the diagonal), $\mathbf{u}_i \equiv \mathbf{U}_{:,i}$ corresponds to the i -th column in $\mathbf{U}$ (i.e., the i -th POD mode), and $\gamma_i \equiv \mathbf{\Gamma}_{:,i}$ is the i -th column of $\mathbf{\Gamma}$ (i.e., the i -th POD coefficient).

A characteristic of SVD is that the singular values are arranged in descending order according to their energy contribution, such that $\sigma_i \geq \sigma_j$ for $i > j$. This ordering implies that the i -th

POD mode captures larger-scale structures in the flow dynamics compared to the j -th POD mode when $i > j$.

This hierarchical structure enables the truncation of POD modes, allowing the retention of only the most relevant dynamics while discarding less significant spatio-temporal scales, which is useful in many practical scenarios (Jiménez, 2018). Furthermore, from (2.11), it is evident that the mode amplitudes in (2.2) for the dataset $\tilde{\mathbf{D}}$ can be expressed as the product of the singular values and the POD coefficients, as follows,

$$a_i(t) = \sigma_i \gamma_{t,i}. \quad (2.12)$$

Here $\gamma_{t,i} \equiv \mathbf{\Gamma}_{t,i}$ represents the t -th element of the i -th POD coefficient γ_i . In practice, applying SVD to a dataset is straightforward using Python programming language libraries such as NumPy (Harris et al., 2020) or PyTorch (Paszke et al., 2019), which enables the implementation of SVD with a single line of code.

As shown in equation (2.12), the temporal information is now encoded in the POD coefficients, γ_i , which are represented as vectors. Consequently, the DL model used to predict the time-ahead evolution of these coefficients must be designed to accept vector inputs.

Note that, while the DL model could alternatively be fitted with the POD amplitudes instead of the coefficients, since both encode the temporal evolution of the modes, a good practice before training DL models is to normalize inputs through standardization (Goodfellow et al., 2016). However, standardizing the POD amplitudes directly would remove the physical significance of weighting the POD coefficients by their corresponding singular values, as expressed in eq. (2.12).

For this reason, it is preferable to train the DL model exclusively on the POD coefficients, applying standardization to these coefficients alone. After predicting the future coefficients and reversing the standardization, they can be weighted by the singular values to restore their physical interpretation. This final step ensures that the full-order dynamics can be reconstructed by performing the matrix multiplication defined in eq. (2.10).

2.2.2 Forecasting DL model

Since SVD provides historical data on the POD coefficients, as expressed in equation (2.12), it is possible to predict their future evolution by leveraging past information. This approach eliminates the need to project the full-order equations onto the POD modes to construct the reduced-order system, as in (2.3). As a result, it enables the development of a fully data-driven methodology that integrates SVD with forecasting models. Moreover, as demonstrated in Chapter 3, this methodology can be effectively applied to forecast datasets derived from experimental data, where the governing equations may be challenging to obtain.

Following the methodology established in similar studies (Abadía-Heredia et al., 2022; Abadía-Heredia, Corrochano, et al., 2024; Besabe et al., 2025; Corrochano et al., 2023; Mohan and Gaitonde, 2018; Parish and Carlberg, 2020; Pawar et al., 2019; Regazzoni et al., 2019; Xiao et al., 2019; Xu et al., 2022), in this chapter we study the performance of a DL model ($\mathcal{M}$) to

forecast the POD coefficients, and consequently the mode amplitudes.

The architecture of this model must be designed to capture the temporal correlations within a past sequence of length m $\{\gamma^{s-m+1}, \dots, \gamma^s\}$ of the first k POD coefficients, corresponding to the k most energetic modes retained after truncation, and to extrapolate this information for predicting the future state of the coefficients, $\{\gamma^{s+1}\}$. Here, $\gamma^s \equiv \mathbf{\Gamma}_{s,1:k}$ represents the s -th state (i.e., s -th snapshot) of the first k retained coefficients.

In this aim, a Long Short-Term Memory (LSTM) network (Hochreiter and Schmidhuber, 1997) is employed. This DL architecture is characterized by its use of three gates: the input gate, the forget gate and the output gate. These gates regulate the flow of information, of the input sequences, through the network by selectively adding new information (input gate), removing irrelevant information (forget gate), or allowing information to pass through to the next cell (output gate).

In this work, the input gate is represented by $\mathbf{i}$, the forget gate by $\mathbf{f}$ and the output gate by $\mathbf{o}$. As well, the cell input is represented by $\mathbf{c}_i$, the cell output is given by $\mathbf{c}_o$ and the cell state is denoted by $\mathbf{c}$. The procedure to compute the gates and states are as follows,

$$\mathbf{f}_j = \hat{\sigma}(\mathbf{W}_f[\mathbf{h}_{j-1}, \mathbf{x}_j] + \mathbf{b}_f) \quad (2.13)$$

$$\mathbf{i}_j = \hat{\sigma}(\mathbf{W}_i[\mathbf{h}_{j-1}, \mathbf{x}_j] + \mathbf{b}_i) \quad (2.14)$$

$$\tilde{\mathbf{c}}_t = \tanh(\mathbf{W}_c[\mathbf{h}_{j-1}, \mathbf{x}_j] + \mathbf{b}_c) \quad (2.15)$$

$$\mathbf{c}_j = \mathbf{f}_j \cdot \mathbf{c}_{j-1} + \mathbf{i}_j \cdot \tilde{\mathbf{c}}_j \quad (2.16)$$

$$\mathbf{o}_j = \hat{\sigma}(\mathbf{W}_o[\mathbf{h}_{j-1}, \mathbf{x}_j] + \mathbf{b}_o) \quad (2.17)$$

$$\mathbf{h}_j = \mathbf{o}_j \cdot \tanh(\mathbf{c}_j) \quad (2.18)$$

Where $\mathbf{W}$ and $\mathbf{b}$ are the weights and biases for each gate, respectively, and $\tilde{\mathbf{c}}$ is the updated cell state. The activation function is represented by $\hat{\sigma}(\cdot)$, which is typically a Sigmoid. Also, the operator $\cdot$ in equations (2.16) and (2.18) represents the Hadamard product.

To enhance the model's flexibility, after applying the LSTM and obtaining the final state, representing the time-ahead prediction, two multi-layer perceptron (MLP) layers are subsequently applied. This process yields the final forecast of the mode amplitudes, $\{\gamma^{t+1}\}$.

The training of this model is done using the T_1 available coefficients $\{\gamma^t\}_{t=1}^{T_1}$, which were obtain by applying SVD to the dataset $\tilde{\mathbf{D}}$ as in (2.10). After training, predictions of the future amplitudes $\{\gamma^t\}_{t=T_1+1}^T$ are performed following an iterative procedure where predictions are taken as new inputs for the model, yielding to an autoregressive model. Note, the solutions $\mathbf{v}^k(x, t)$ are constructed using (2.10). A visual depiction of the POD-DL ROM is shown in Fig. 2.1.

2.3 Results

The ROM presented in this chapter has been evaluated on a dataset describing the three-dimensional wake of a circular cylinder. This dataset has been generated numerically and

includes information from the initial transient phase of the simulations (Le Clainche et al., 2018). This section presents the results obtained by using the POD-DL ROM to predict the temporal evolution of this dataset, with different normalization techniques implemented to assess the model’s sensitivity to normalization.

The speed-up factor quantifies the computational efficiency of the predictive ROM by comparing the total time required to train and generate predictions with the time needed for a full numerical simulation. It is defined as the ratio of the combined training time (t_{tr}) and prediction time (t_{pred}) to the computational time (t_{solv}) required by the numerical solver to simulate the predicted interval.

$$\text{Speed-up} = \frac{t_{solv}}{t_{pred} + t_{tr}}. \quad (2.19)$$

A higher speed-up value indicates a more computationally efficient ROM compared to the full-order numerical solver.

The governing equations describing the movement of the viscous flow, incompressible and Newtonian, for the wake of a circular cylinder, are the Navier-Stokes equations, written in non-linear, non-conservative form as,

$$\nabla \cdot \mathbf{v} = 0, \quad (2.20)$$

$$\frac{\partial \mathbf{v}}{\partial t} + (\mathbf{v} \cdot \nabla) \mathbf{v} = -\nabla \mathbf{p} + \frac{1}{Re} \Delta \mathbf{v}. \quad (2.21)$$

Here $\mathbf{v}$ and $\mathbf{p}$ denote the non-dimensional velocity vector and pressure, respectively. The Reynolds number, Re , is defined as $Re = VL/\nu$, where ν is the kinematic viscosity of the fluid, V is the characteristic flow velocity and L is the characteristic length. As usual, the governing equations are non-dimensionalized using L as the reference length and L/V as the characteristic time scale.

As explained in the previous section, the POD-DL model is a data-driven approach that operates on datasets composed of discrete snapshots. To prevent potential confusion, from now on, the variable t is reserved to denote the time step within the continuous temporal domain (i.e., $t \in \mathcal{P}_t$), while the variable t_s is used to refer to the index of the snapshots that constitute the dataset and discretize the temporal domain $\mathcal{P}_t$.

To assess the accuracy of predictions, two key metrics are employed: the root mean square error (RMSE) and the structural similarity index measure (SSIM). These metrics are primarily used for this flow, where the non-turbulent dynamics allow for a direct comparison between predicted and reference solutions, in contrast to turbulent flows where the stochastic nature of the dynamics shifts the objective from exact trajectory matching to statistical consistency, as shown in Chapter 3. In this context, accuracy is evaluated based on the similarity between snapshots obtained from numerical simulations and those predicted using the POD-DL model. On the one hand, the RMSE, computed at each time-step, is defined as follows,

$$\text{RMSE}(t_S) = \sqrt{\frac{\sum_{x=1}^N (\mathbf{v}_s - \hat{\mathbf{v}}_s)^2}{N}}, \quad (2.22)$$

where $\mathbf{v}_s$ and $\hat{\mathbf{v}}_s$ denote the s -th ground truth and predicted snapshot, respectively. The scalar $N = N_x * N_y * N_z$, as before, denotes the number of discretization points in the spatial dimension.

On the other hand, the SSIM is used to measure the degree of similarity between two snapshots based on their structural information, as follows,

$$\text{SSIM}(t_S) \equiv \text{SSIM}(\mathbf{v}_s, \hat{\mathbf{v}}_s) = \frac{(2\mu_{\mathbf{v}_s}\mu_{\hat{\mathbf{v}}_s} + c_1)(2\sigma_{\mathbf{v}_s\hat{\mathbf{v}}_s} + c_2)}{(\mu_{\mathbf{v}_s}^2 + \mu_{\hat{\mathbf{v}}_s}^2 + c_1)(\sigma_{\mathbf{v}_s}^2 + \sigma_{\hat{\mathbf{v}}_s}^2 + c_2)}, \quad (2.23)$$

where $\mu_{\mathbf{v}_s}$ and $\mu_{\hat{\mathbf{v}}_s}$ represent the mean values of the ground truth and predicted snapshots, respectively. Similarly, $\sigma_{\mathbf{v}_s}$ and $\sigma_{\hat{\mathbf{v}}_s}$ denote their corresponding standard deviations. Additionally, $\sigma_{\mathbf{v}_s\hat{\mathbf{v}}_s}$ represents the covariance between the two snapshots, while c_1 and c_2 are fixed constants introduced to prevent numerical instability.

The POD-DL model is designed to perform autoregressive predictions over a long time horizon for both datasets. As demonstrated in Chapter 3, this model stands out from other DL approaches due to its ability to achieve accurate predictions with a significantly smaller number of training samples. Consequently, this chapter presents the predictions obtained when the POD-DL model is trained with a minimal amount of data, as summarized in Tab. 2.1.

Table 2.1: Distribution of training and testing samples used for the POD-DL model.

Dataset	Training	Testing
Laminar	100	200

The DL model used in this chapter is trained with the following parameters: (a) optimization using the Adam algorithm (D. P. Kingma and Ba, 2017), with default parameter values ($\alpha = 0.001$) for the learning rate, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$; see details in D. P. Kingma and Ba, 2017); and (b) mini-batch gradient descent with a batch size of 12 and a training duration of 300 epochs. The loss function used to train the model is the usual MSE, defined as follows,

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (\mathbf{v}_i - \hat{\mathbf{v}}_i), \quad (2.24)$$

where n represents the total number of samples in the training set, $\mathbf{v}_i$ corresponds to the i -th ground truth sample in the training set and $\hat{\mathbf{v}}_i$ is the prediction from the model. One of the objectives of this chapter is to assess the sensitivity of the POD-DL ROM to different

normalization techniques. These normalizations can be applied at various stages within the ROM.

On one hand, normalization can be performed before applying SVD, which is particularly useful in combustion problems where variables may differ in scale by several orders of magnitude. On the other hand, normalization can also be applied directly to the POD coefficients computed via SVD. While the latter directly influences the training process of the DL model, the former also plays a role, as it affects both the number of POD modes retained after truncation and the dynamics of the coefficients to be predicted, both of which vary depending on the chosen normalization technique.

The two normalizations performed before and after applying SVD will be identified for the rest of the thesis as Norm₁ and Norm₂, respectively. In this study each normalization can be applied using 3 different techniques described in Tab. 2.3.

In this table, the expression $\min_{j_2, j_3, j_4, j_5} \dot{\mathbf{D}}_{:,j_2, j_3, j_4, j_5}$ indicates that the minimum operator is computed over the second, third, fourth and fifth dimensions of the dataset $\dot{\mathbf{D}}$, while retaining the first dimension. Consequently, $\min_{j_2, j_3, j_4, j_5} \dot{\mathbf{D}}_{:,j_2, j_3, j_4, j_5}$ results in a vector with C elements. The same is applied to the maximum (max), mean and standard deviation (std) operators, as well as to the POD coefficients ($\mathbf{\Gamma}_{:,1:7}$).

Table 2.3: Normalizations applied to the POD-DL model, where each of them utilize different techniques. One of the normalization is applied directly on the dataset, before applying SVD, and the other on the POD coefficients, after SVD.

Normalization	Technique	Expression
Norm ₁ = $\frac{\dot{\mathbf{D}} - n_{11}}{n_{12}}$	None	$n_{11} = 0$ $n_{12} = 1$
	Range	$n_{11} = \min_{j_2, j_3, j_4, j_5} \dot{\mathbf{D}}_{:,j_2, j_3, j_4, j_5}$ $n_{12} = \left(\max_{j_2, j_3, j_4, j_5} \dot{\mathbf{D}}_{:,j_2, j_3, j_4, j_5} \right) - n_{11}$
	Standardization	$n_{11} = \text{mean}_{j_2, j_3, j_4, j_5} \dot{\mathbf{D}}_{:,j_2, j_3, j_4, j_5}$ $n_{12} = \text{std}_{j_2, j_3, j_4, j_5} \dot{\mathbf{D}}_{:,j_2, j_3, j_4, j_5}$
Norm ₂ = $\frac{\mathbf{\Gamma}_{:,1:k} - n_{21}}{n_{22}}$	None	$n_{21} = 0$ $n_{22} = 1$
	Range	$n_{21} = \min_{j_1} \mathbf{\Gamma}_{j_1, 1:k}$ $n_{22} = \left(\max_{j_1} \mathbf{\Gamma}_{j_1, 1:k} \right) - n_{21}$
	Standardization	$n_{21} = \text{mean}_{j_1} \mathbf{\Gamma}_{j_1, 1:k}$ $n_{22} = \text{std}_{j_1} \mathbf{\Gamma}_{j_1, 1:k}$

The next subsection presents the predictions obtained from the POD-DL model when applied to the dataset describing a flow past a circular cylinder, with all 9 combinations of the different

normalizations shown in Tab. 2.3. Additionally, it provides a detailed explanation of the training process, including key considerations such as selecting the appropriate number of modes to retain during SVD. This selection depends on the specific characteristics of the dataset and plays a crucial role in model performance.

2.3.1 Three-dimensional wake of a flow past a circular cylinder.

The dataset used to test the POD-DL model, in this chapter, correspond to the three-dimensional wake of a flow past a circular cylinder. This type of flow is a well-established benchmark problem in fluid dynamics, widely used to illustrate various flow phenomena and applications. The wake dynamics of a circular cylinder are governed by variations in the Reynolds number, as previously defined $Re = VL/\nu$, where D is the cylinder diameter and V is the incoming free stream velocity.

At low Reynolds numbers, the flow remains steady. The first bifurcation occurs at $Re \simeq 46$, where a Hopf bifurcation induces two-dimensional flow oscillations in the cylinder wake, leading to the formation of a von Kármán vortex street (Jackson, 1987). As the Reynolds number increases, the flow transitions to three-dimensional structures for specific values of the spanwise length $L_z = 2\pi/\beta$, where β represents the spanwise wavenumber (Barckley and Henderson, 1996).

Specifically, this dataset, presented in Le Clainche et al., 2018, corresponds to the three-dimensional wake of a circular cylinder at $Re = 280$ within a computational domain of spanwise length $L_z = 6.99$, where the solution exhibits periodic behavior. The numerical dataset was generated using the open-source code Semtex (Blackburn and Henderson, 1999), which solves the fully non-linear, incompressible Navier–Stokes equations (2.20)-(2.21).

The dimensions and boundary conditions of the computational domain follow those established by Barckley and Henderson, 1996, with the cylinder diameter set to $D = 1$, serving as the reference length scale for spatial normalization.

The imposed boundary conditions are as follows: Dirichlet conditions for velocity, with $V_x = 1$ and $V_y = 0$ (streamwise and normal components, respectively), and Neumann conditions for pressure at the inlet, upper and lower boundaries. At the outlet, Neumann conditions are applied to velocity, while Dirichlet conditions are imposed on pressure. In the spanwise direction, periodic boundary conditions are enforced with a period L_z , and the equations are discretized using 64 equidistant Fourier collocation planes. The cylinder boundaries are modeled using Dirichlet conditions for velocity ($V_x = V_y = 0$) and Neumann conditions for pressure.

The computational domain, non-dimensionalized by the cylinder diameter ($D = 1$), extends to $L_y = \pm 15$ in the normal direction and $L_x = 15$ upstream and $L_x = 50$ downstream of the cylinder in the streamwise direction. For efficiency, a reduced computational domain focusing on the cylinder wake is used for model construction, with dimensions $L_y = \pm 2$ and $L_x = 10$ in the downstream direction.

The dataset consists of 599 snapshots collected from the beginning of the numerical simulations ($t = t_S = 0$). The snapshots are uniformly spaced in time with a time step of $\Delta t = 1$. As

shown in Fig. 2.2, which depicts the evolution of the spatially averaged streamwise ($\bar{v}_x$) and normal ($\bar{v}_y$) velocity components, a clear transient phase is observed between $t_S = 0$ and $t_S = 100$. However, the simulation does not reach convergence until at least $t_S = 500$ (see details in Le Clainche et al., 2018).

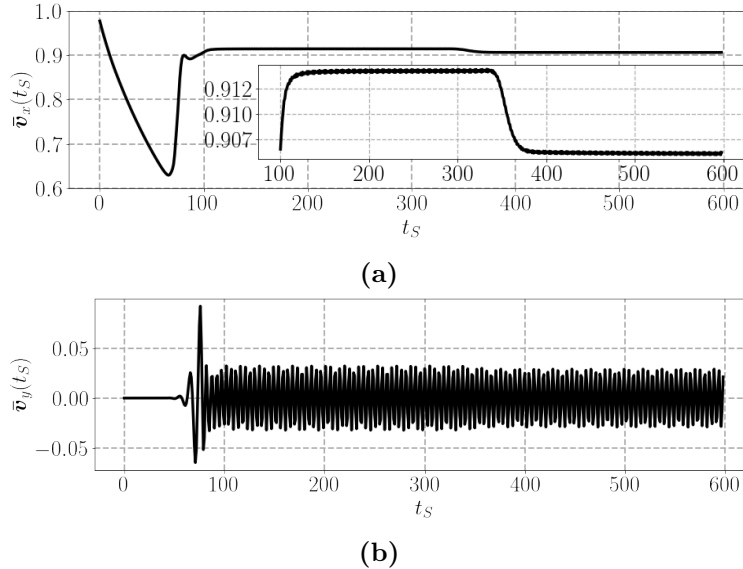


Figure 2.2: Temporal evolution of the spatially averaged streamwise (a) and normal (b) velocity components. Here, t_S denotes the snapshot index rather than the actual time t .

The first step of the POD-DL framework is to apply proper orthogonal decomposition (POD), or in this case singular value decomposition (SVD), to the dataset in order to compute the POD coefficients associated with the temporally varying flow structures, as expressed in equation (2.10). However, to prevent data leakage, SVD should not be applied to the entire dataset but only to the training samples.

As indicated in Tab. 2.1, for this flow, only the first 100 snapshots are used for training the POD-DL model. However, since the first 100 snapshots ($t_S \in [0, 99]$) correspond to the transient phase, as shown in Fig. 2.2, they are discarded from the training process.

Consequently, the training samples consist of snapshots in the temporal range $t_S \in [100, 199]$, represented as $\mathbf{D}_{:,100:199}$. After computing and subtracting the mean flow from this 100 snapshots, as in equation (2.8), SVD is applied exclusively to this subset. The truncation of the number of modes is based on their cumulative energy, which represents the percentage of total energy captured by the first k modes. This metric is described as follows,

$$E = \frac{\sum_{i=1}^k \lambda_i}{\sum_{i=1}^K \lambda_i}, \quad (2.25)$$

where λ_i is the i -th singular value, K the total number of modes and k the number of retained modes.

The truncation process is typically guided by the decay of singular values. However, as

illustrated in Fig. 2.3, both the rate of decay and the cumulative energy of singular values depend on the normalization (Norm_1) applied before SVD.

The figure demonstrates that the singular values decay much more rapidly when $\text{Norm}_1 = \text{None}$ (a) and $\text{Norm}_1 = \text{Range}$ (b) compared to $\text{Norm}_1 = \text{Standardization}$ (c). This indicates that in the latter case, a larger number of POD modes must be retained to preserve the same level of dynamical information as in the other two cases. A similar trend is observed in the cumulative energy plots: for $\text{Norm}_1 = \text{None}$ (d) and $\text{Norm}_1 = \text{Range}$ (e), the first few modes capture a greater proportion of the total energy than in the case of $\text{Norm}_1 = \text{Standardization}$ (f).

Since the snapshots composing the training set ($\mathbf{D}_{:,100:199}$) belong to a laminar regimen, it is of interest to retain at least 90% of the overall energy. Table 2.5 shows the number of POD modes retained for each normalization technique and their respective cumulative energy.

Table 2.5: Number of POD modes retained based on the normalization technique choose for Norm_1 .

Norm_1	POD modes retained	Cumulative energy (%)
None	6	90.5%
Range	4	91.3%
Standardization	6	90.2%

For $\text{Norm}_1 = \text{None}$ and $\text{Norm}_1 = \text{Standardization}$, only the first 6 POD modes are required to retain approximately 90% of the overall energy, indicating that predicting the temporal evolution of these six modes is sufficient to capture the dominant flow dynamics. Consequently, in these cases, the DL model is trained exclusively on the first six POD coefficients. The shortest truncation is done for $\text{Norm}_1 = \text{Range}$, where the number of retained modes decreases to 4.

To illustrate the significant dimensionality reduction achieved by applying POD via SVD, consider the original training dataset, which has a dimensionality of $[2 \times 100 \times 40 \times 64 \times 100]$, where 100 high-dimensional snapshots are included. Training a DL model directly on such a dataset would require specialized architectures, as discussed in Chapter 3 and Chapter 5.

However, after applying POD and retaining the first 6 POD coefficients ($\mathbf{\Gamma}_{:,1:6}$) for training, the dataset's dimensionality is reduced to $\dim(\mathbf{\Gamma}_{:,1:6}) = [100 \times 6]$. This transformation effectively reduces the data representation from high-dimensional snapshots ($[2 \times 100 \times 40 \times 64]$) to a significantly lower-dimensional vector representation ($[100 \times 6]$), thereby facilitating a more computationally efficient learning process. This reduction allows also for the use of less complex DL architectures with a smaller number of trainable parameters, improving both training efficiency and model interpretability (Lundberg and Lee, 2017), while making it less prone to overfitting risk.

Before training the DL model, the training set must be structured to distinguish between input samples and target outputs. Since the objective is to predict the future evolution of

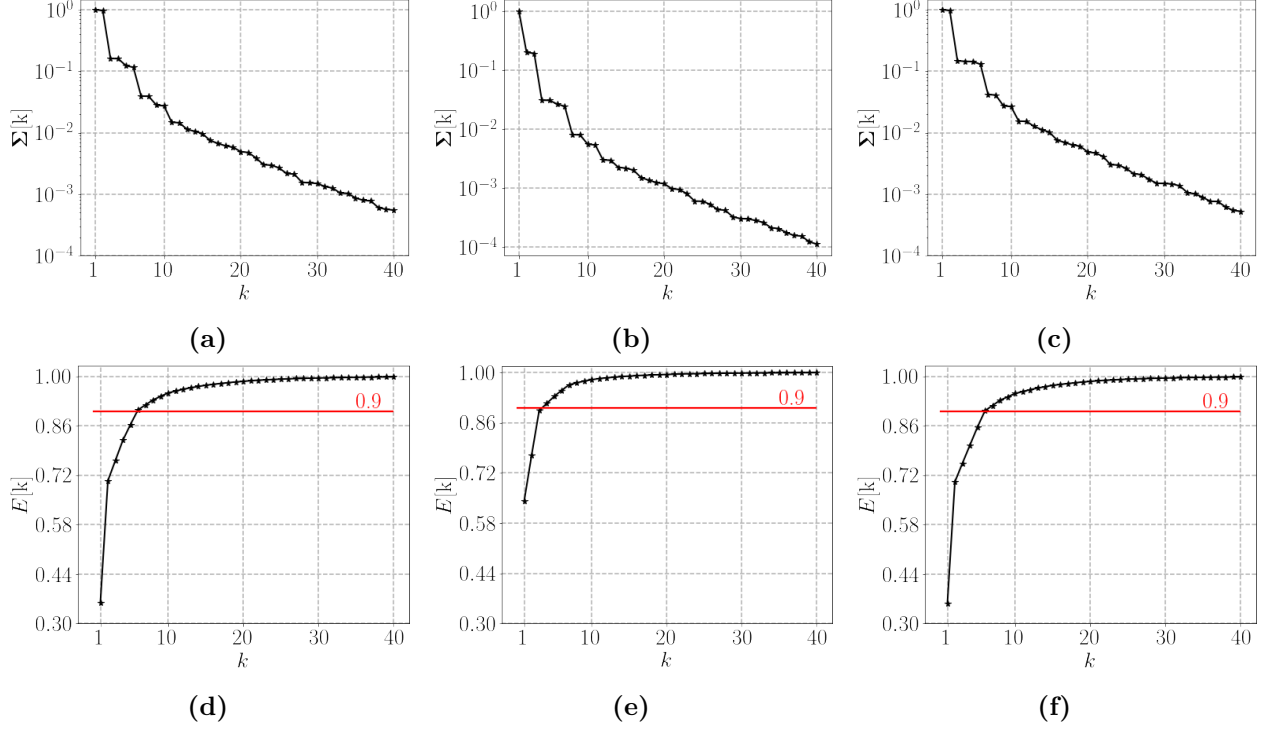


Figure 2.3: Top row shows the singular value decay for $\text{Norm}_1 = \text{None}$ (a), $\text{Norm}_1 = \text{Range}$ (b) and $\text{Norm}_1 = \text{Standardization}$ (c). Bottom row shows the cumulative energy for $\text{Norm}_1 = \text{None}$ (d), $\text{Norm}_1 = \text{Range}$ (e) and $\text{Norm}_1 = \text{Standardization}$ (f).

flow dynamics, the dataset is transformed into a series of overlapping windows. Each window consists of a past sequence of samples ($\{\gamma^{s-m+1}, \dots, \gamma^s\}$) serving as the model input, and the subsequent time-ahead sample ($\{\gamma^{s+1}\}$) as the target.

This transformation is achieved using a technique known as the rolling-window method, which is described in more detail in Appendix A.

For this dataset, the input windows consist of $m = 10$ previous coefficients. After applying the rolling-window technique to the entire training set ($\mathbf{\Gamma}_{:,1:k}$), the DL model ($\mathcal{M}$) is ready for training. Once trained, the model can generate 200 time-ahead predictions in an autoregressive manner, as follows,

$$\hat{\gamma}^{200} = \mathcal{M}(\{\gamma^{190}, \dots, \gamma^{199}\}), \quad (2.26)$$

$$\hat{\gamma}^{201} = \mathcal{M}(\{\gamma^{191}, \dots, \gamma^{199}, \hat{\gamma}^{200}\}), \quad (2.27)$$

$$\vdots$$

$$\hat{\gamma}^{399} = \mathcal{M}(\{\hat{\gamma}^{389}, \dots, \hat{\gamma}^{398}\}). \quad (2.28)$$

Here $\gamma^s \equiv \mathbf{\Gamma}_{s,1:k}$ and $\hat{\gamma}^s$ denote the s -th state of the ground truth and predicted POD coefficients, respectively. The variable $k = \{4, 6\}$ depending on the normalization technique applied for Norm_1 .

Note from equation (2.10), that to obtain the ground truth POD coefficients for snapshots in $t_S \in [200, 399]$, is enough to perform a matrix multiplication as follows,

$$\mathbf{\Gamma}_{1:k,200:399}^* = \mathbf{\Sigma}^{-1} \mathbf{U}^* \tilde{\mathbf{D}}_{:,200:399}, \quad (2.29)$$

where $\mathbf{\Sigma}^{-1}$ and $\mathbf{U}$ are the singular values and POD modes computed after applying SVD to the training snapshots $\tilde{\mathbf{D}}_{:,100:199}$, respectively. The superscript $()^{-1}$ denotes the inverse operator, and since $\mathbf{U}$ is an orthonormal matrix the inverse is the conjugate transpose.

As stated above, before training the DL model the POD coefficients $(\mathbf{\Gamma}_{:,1:k})$ are normalized using Norm₂, which may affect the training of the model. Figure 2.4 compares the evolution of the first, second and fourth unnormalized ground truth POD coefficients, computed by equation (2.29), and unnormalized predictions from the DL model, computed by equations (2.26 - 2.28), for all combinations of Norm₁ and Norm₂.

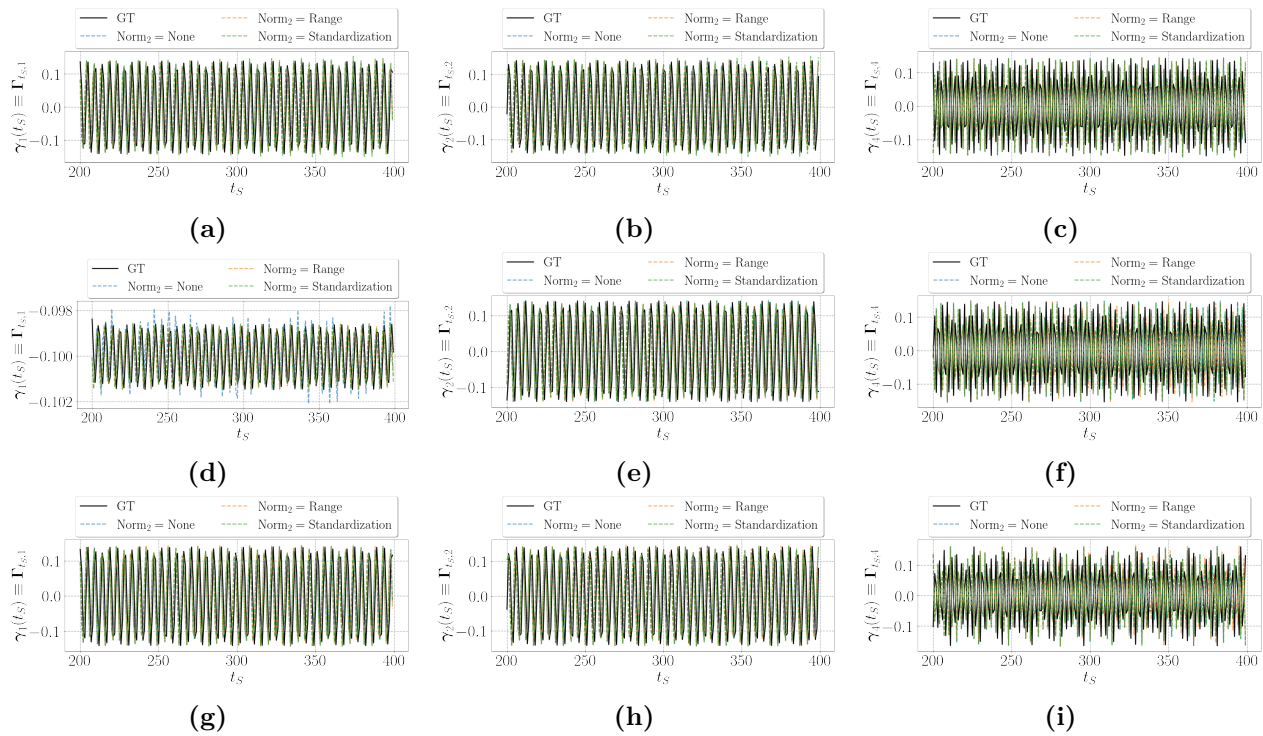


Figure 2.4: The top row shows the comparison of the first (γ_1 ; a), second (γ_2 ; b) and fourth (γ_4 ; c) POD coefficients derived from ground truth data (computed according to eq. (2.29)) and obtained from POD-DL predictions across different Norm₂ scaling strategies, with Norm₁ = None. The middle row shows the corresponding first (d), second (e) and fourth (f) coefficients when Norm₁ = Range. The bottom row displays the first (g), second (h) and fourth (i) coefficients for Norm₁ = Standardization.

Once the 200 time-ahead POD coefficients, $\{\hat{\gamma}^{200}, \dots, \hat{\gamma}^{399}\}$, have been computed, the corresponding predicted snapshots can be reconstructed by reversing the SVD and adding the mean flow, as follows,

$$\hat{\hat{\mathbf{D}}}_{:,200:399} = \mathbf{U} \Sigma \hat{\mathbf{\Gamma}}_{1:k,200:399}^* \quad (2.30)$$

$$\hat{\mathbf{D}}_{:,200:399} = \hat{\hat{\mathbf{D}}}_{:,200:399} + \bar{\mathbf{D}}. \quad (2.31)$$

Here $\bar{\mathbf{D}}$ represents the mean flow computed from the training snapshots, as defined in equation (2.8). The next step is to reverse the normalization Norm_1 , followed by reshaping the matrix $\hat{\hat{\mathbf{D}}}_{:,200:399}$ to recover the original tensor format $\hat{\mathbf{D}}_{:,200:399}$.

Note that both the ground truth ($\hat{\mathbf{D}}_{:,200:399}$) and predicted ($\hat{\hat{\mathbf{D}}}_{:,200:399}$) tensors represent the velocity field of the fluid. However, the vorticity field is used to evaluate the accuracy of the predictions through the previously defined metrics: RMSE (2.22) and SSIM (2.23). The vorticity field is computed as follows,

$$\boldsymbol{\omega} = \nabla \times \mathbf{v} = \left(\frac{\partial \mathbf{v}_y}{\partial x} - \frac{\partial \mathbf{v}_x}{\partial y} \right). \quad (2.32)$$

Here, the operator $\times$ denotes the outer product. Similar to the velocity field, the vorticity field is also composed of snapshots. To assess the accuracy of the predictions, both RMSE and SSIM are employed to track the temporal evolution of the prediction errors. These results are presented in Fig. 2.5 for all combinations of the normalization strategies Norm_1 and Norm_2 , as these influence the truncation of POD modes and, consequently, the dimensionality of the training set for the DL model. This, in turn, affects model training and ultimately the accuracy of the results.

Finally, selected snapshots of the ground truth and predicted vorticity fields are compared for different combinations of the normalization strategies Norm_1 and Norm_2 , evaluated at two representative time instances. Figure 2.6 shows the comparison of predicted and ground truth vorticity at snapshots 50 and 190 for the normalization strategy $\text{Norm}_1 = \text{None}$ and $\text{Norm}_2 = \text{None}$. A similar comparison is presented in Fig. 2.7 for the case $\text{Norm}_1 = \text{Range}$ and $\text{Norm}_2 = \text{Standardization}$, and in Fig. 2.8 for $\text{Norm}_1 = \text{Standardization}$ and $\text{Norm}_2 = \text{Range}$.

2.4 Discussion

This chapter introduces the ROM referred to as POD-DL, which integrates POD with DL techniques to perform temporal predictions of flow dynamics. This ROM constitutes one of the core contributions of this thesis, as it is also used in the rest of the thesis. This approach is conceptually motivated by Galerkin projection methods, which reduce the full-order system by projecting the governing equations onto an orthonormal basis formed by the POD modes. This projection yields a reduced system of equations governing the evolution of the POD amplitudes, which can then be solved numerically to approximate, together with the POD modes as shown in equation (2.2), the original flow dynamics.

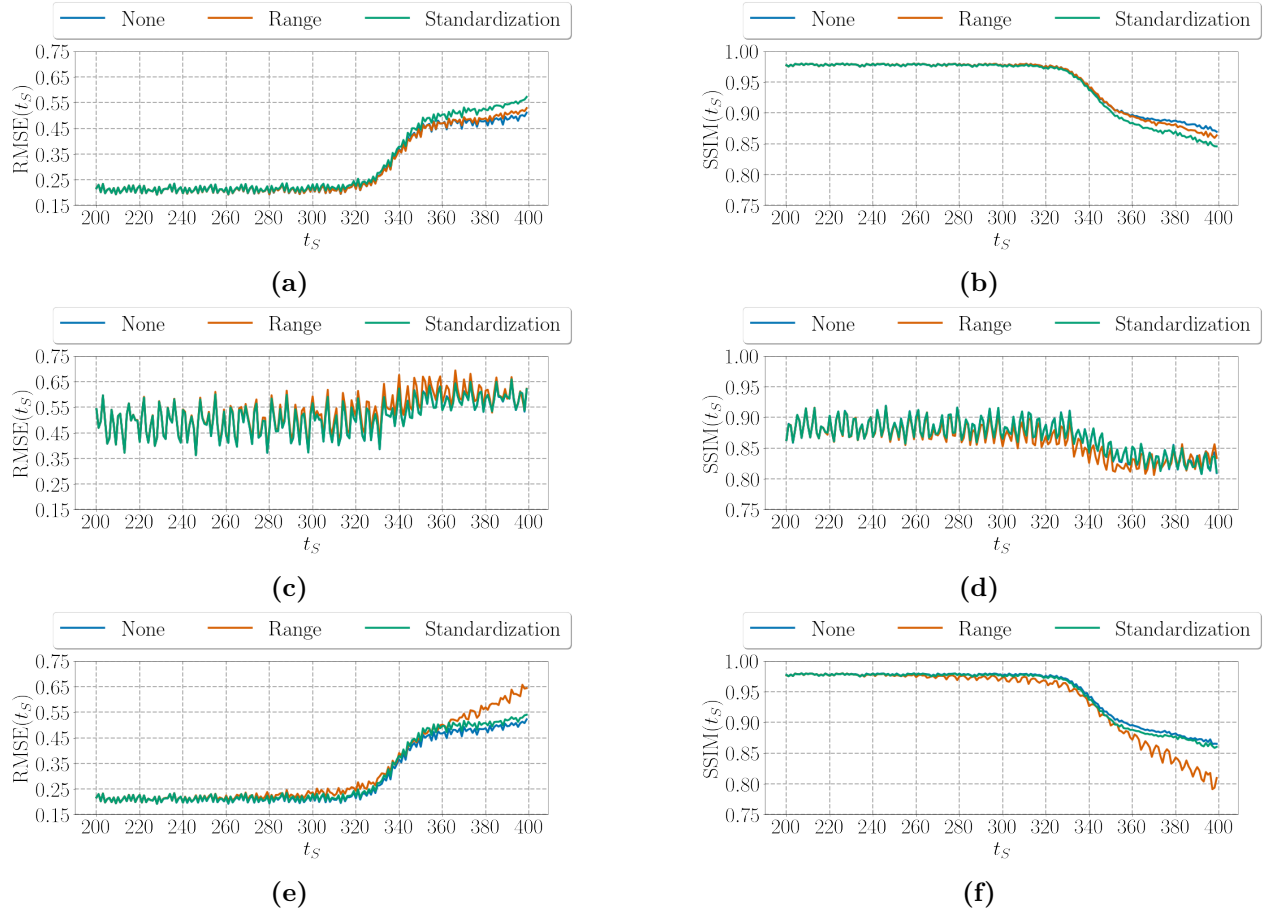


Figure 2.5: The top row shows the RMSE (a) and SSIM (b) obtained across different Norm_2 scaling approaches, with $\text{Norm}_1 = \text{None}$. The middle row shows the corresponding RMSE (c) and SSIM (d) results when $\text{Norm}_1 = \text{Range}$. The bottom row displays the RMSE (e) and SSIM (f) metrics for $\text{Norm}_1 = \text{Standardization}$.

In contrast to the Galerkin approach, the POD-DL model utilizes the POD modes without projecting the governing equations. Instead, it employs a data-driven strategy, using a DL architecture to predict the temporal evolution of the POD coefficients. These coefficients are used to compute the amplitudes of the POD modes, following equation (2.12), and thus provide an approximation of the solution to the governing equations.

As a data-driven model, POD-DL operates on tensors composed of flow snapshots that represent the underlying dynamics. However, unlike purely DL-based models, and as demonstrated in Section 2.3, this hybrid approach enables substantial dimensionality reduction through POD. It is not necessary to retain all POD modes; only the most energetic ones are preserved (Jiménez, 2018), adhering to the same foundational principles as Galerkin-based methods.

POD is implemented via singular value decomposition (SVD), also known as the snapshot method, which enables the decomposition of the snapshot tensor into its corresponding POD modes and coefficients. The temporal evolution of the coefficients is then predicted using a DL model.

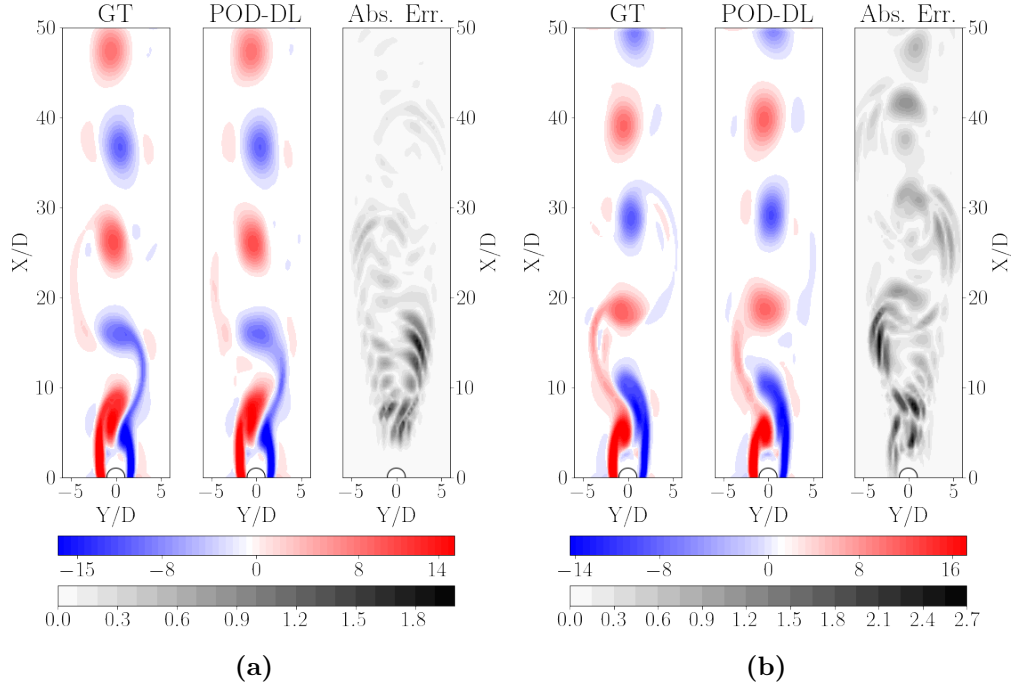


Figure 2.6: Comparison of ground truth (left column) and predicted (middle column) vorticity field for snapshots 50 (a) and 190 (b), with $\text{Norm}_1 = \text{None}$ and $\text{Norm}_2 = \text{None}$. The right column in both figures shows the absolute error between the predicted and ground truth vorticity.

In data-driven modeling approaches, normalization often plays a critical role. Accordingly, this chapter investigates the impact of normalization when applied at two distinct stages: prior to SVD (on the raw dataset) and after SVD (on the POD coefficients). The first normalization directly influences the resulting POD modes and coefficients, as the database may represent the dynamics of variables with markedly different physical scales. For instance, in fluid dynamics, the dataset may contain multiple velocity components, while in combustion problems, it may include distinct chemical species. Without normalization, these scale disparities can lead to a disproportionate energy distribution, potentially necessitating the retention of a large number of POD modes to capture the dominant dynamics.

The second normalization is applied directly to the POD coefficients, which serve as the training input to the DL model. As in the previous case, the coefficients may vary in scale by several orders of magnitude. If left unnormalized, these discrepancies can lead to gradient explosion or vanishing during backpropagation, causing instability in training, increased convergence time, or failure to converge altogether. For this reason, normalization of the training data is essential.

In this chapter, the POD-DL model is tested on a dataset representing the wake flow past a circular cylinder. The model is trained using 100 snapshots and then employed to predict the subsequent 200 time steps, in an autoregressive manner as shown in equations (2.26 – 2.28). In parallel, different combinations of the two normalization strategies applied to the model are examined, they are listed in Tab. 2.3. This analysis aims to assess both the model’s

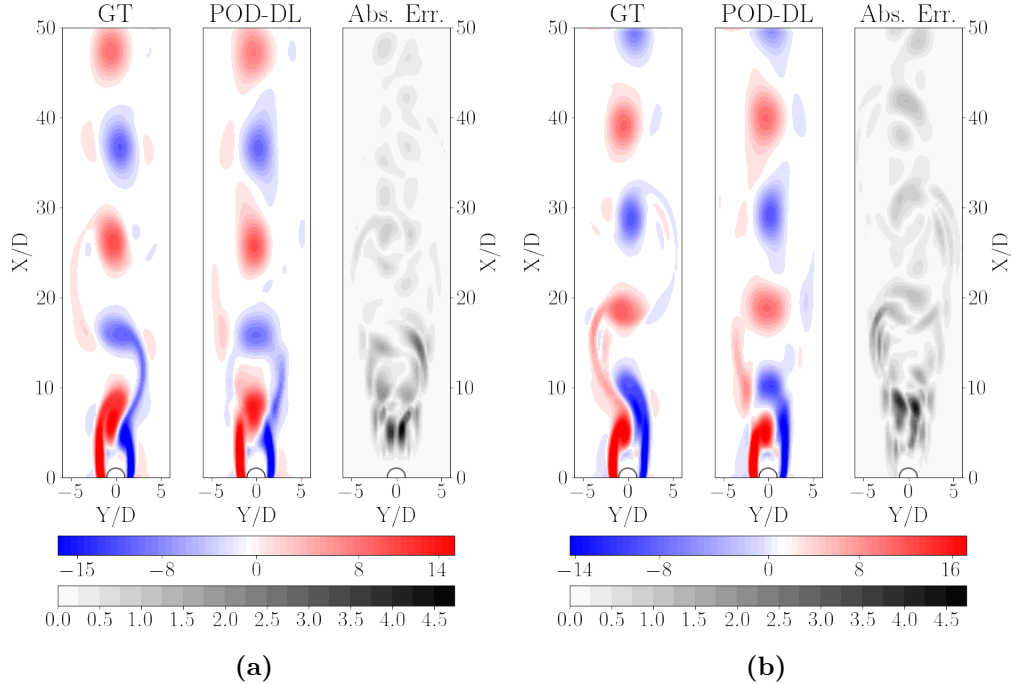


Figure 2.7: Comparison of ground truth (left column) and predicted (middle column) vorticity field for snapshots 50 (a) and 190 (b), with $\text{Norm}_1 = \text{Range}$ and $\text{Norm}_2 = \text{Standardization}$. The right column in both figures shows the absolute error between the predicted and ground truth vorticity.

stability in generating long-range temporal predictions and its robustness with respect to the chosen normalization schemes.

As illustrated in Fig. 2.4, which presents a comparison between the temporal evolution of the four most energetic predicted and actual POD coefficients, the POD-DL model demonstrates the ability to produce stable predictions over time, irrespective of the normalization strategy applied. Notably, even for the less energetic modes, characterized by noisier dynamics, the model continues to yield accurate forecasts.

These findings are consistent with those reported in Fig. 2.5, which compares the actual and predicted vorticity fields, reconstructed from the predicted POD coefficients. The prediction errors in this figure are quantified using the RMSE (2.22) and SSIM (2.23) metrics. Again, the results indicate that the model's predictions remain stable over time across all normalization scenarios. However, some differences are observed that may not be evident from the POD coefficients alone, as even minor discrepancies in the coefficients can amplify upon reconstruction using the POD modes.

This can be seen in Fig. 2.5, where the prediction errors remain within a similar range across all normalization strategies. It is important to note that the RMSE is reported in absolute magnitudes rather than percentages. As such, the error values are relatively small in all cases, particularly when considering that the reference magnitude of the vorticity field is approximately 8, as shown in Fig. 2.6, Fig. 2.7 and Fig. 2.8.

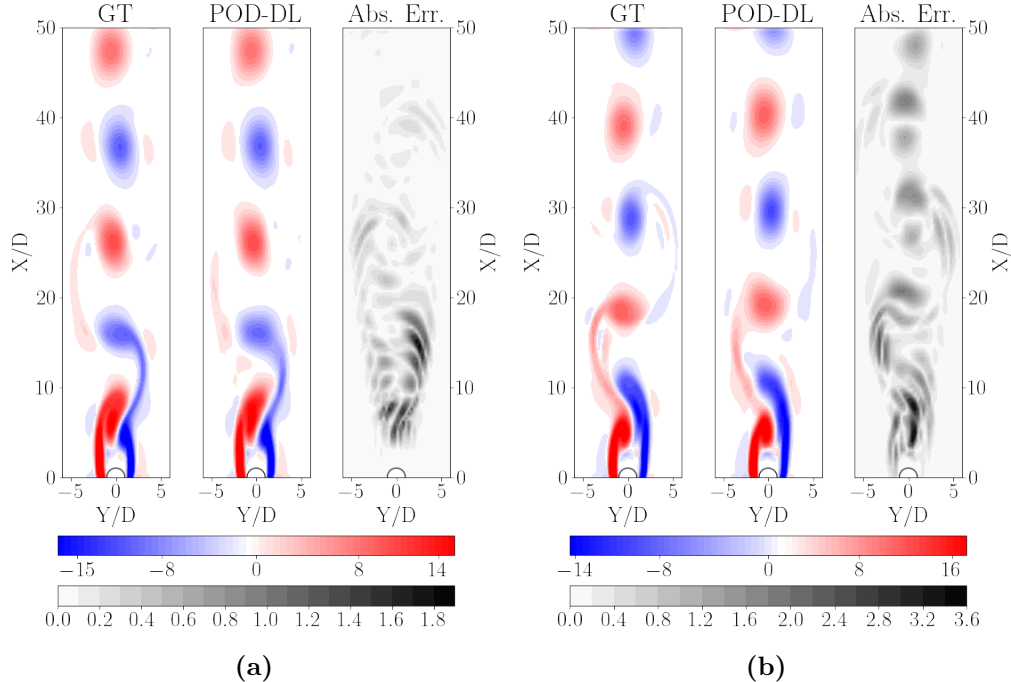


Figure 2.8: Comparison of ground truth (left column) and predicted (middle column) vorticity field for snapshots 50 (a) and 190 (b), with $\text{Norm}_1 = \text{Standardization}$ and $\text{Norm}_2 = \text{Range}$. The right column in both figures shows the absolute error between the predicted and ground truth vorticity.

It is also worth noting that the predictions obtained with $\text{Norm}_1 = \text{Range}$ are slightly less accurate than those from the other configurations, as evidenced by both the RMSE (Fig. 2.5(c)) and SSIM (Fig. 2.5(d)) results. One of the motivations for exploring various combinations of normalization strategies was to assess whether a specific pairing could positively or negatively influence the model’s predictive performance. Although no clearly superior combination was identified, Fig. 2.5 (e) and (f) suggest that the combination $\text{Norm}_1 = \text{Standardization}$ and $\text{Norm}_2 = \text{Range}$ has a slightly adverse effect on the accuracy of the predictions.

However, in all cases, the reconstruction of the vorticity field remains faithful, as shown in Fig. 2.6, Fig. 2.7 and Fig. 2.8, which compare the actual and predicted vorticity fields, along with their corresponding absolute error, at two representative time steps and for different normalization strategies. These figures also reveal that the error observed at later time steps is primarily due to a slight phase shift in the predicted dynamics relative to the actual dynamics, indicating the onset of divergence in the predictions.

This chapter has demonstrated the capability of the POD-DL ROM to perform long-term temporal predictions, as well as its robustness to various normalization techniques, in a canonical fluid dynamics problem: the three-dimensional wake of a flow past a circular cylinder.

In this chapter, as well as in Chapter 3 and Chapter 4, a DL model based on long short-term memory (LSTM) network has been employed. However, the flexibility of the POD-DL framework permits the integration of various DL architectures for the temporal prediction of

POD coefficients, making it adaptable to different modeling needs and complexities.

Finally, it is worth highlighting the computational efficiency of the POD-DL model, both in terms of training and prediction. The experiments, carried out in this chapter, were performed on a hardware equipped with an Intel(R) Core(TM) i7-10700 CPU @ 2.90 GHz, featuring 16 cores and 64 Gb of RAM. Computation times were obtained by averaging multiple runs and are as follows: training the DL model with 100 snapshots over 300 epochs requires approximately 23 seconds, while generating 200 future snapshot predictions takes around 6 seconds.

Considering that the computational time required by the numerical solver to simulate the saturated regime of the three-dimensional flow, computing 200 snapshots, is approximately 160 hours (or 100 hours using 16 processors on a computer with 32 GB of RAM and an Intel i5 processor), the computational speed-up achieved by the POD-DL model is approximately 12413 times, as calculated using Equation (2.19).

The following chapter compares the POD-DL ROM with purely DL-based forecasting models and highlights the enhanced generalization capabilities offered by this hybrid approach.

Chapter 3

Generalization capabilities of the hybrid POD-DL model against fully DL models

In Chapter 2, a hybrid ROM integrating POD and deep learning techniques was introduced as an efficient framework for accelerating numerical simulations by forecasting key dynamical behaviors. The utility of POD in this context is twofold: (1) it mitigates the computational burden by reducing the dimensionality of the input data, transforming high-dimensional snapshots into compact vector representations via POD coefficients, and (2) it enables targeted prediction of dominant flow features by filtering out small-scale, less significant structures. Subsequently, a deep learning model is trained on the temporal evolution of the POD coefficients, enabling the reconstruction of future spatiotemporal flow dynamics from the predicted low-dimensional representations.

However, as outlined in the Introduction (Chapter 1), many contemporary models for forecasting flow dynamics rely entirely on deep learning architectures, with autoencoders being particularly prominent (Han et al., 2019; Hasegawa et al., 2020; Nakamura et al., 2021; Temirchev et al., 2020). These models are typically composed of an encoder–decoder structure.

Their popularity stems from the encoder’s ability to decompose the spatial dimension of flow snapshots, whether obtained from numerical simulations or experimental measurements, into dominant spatial features or patterns. This decomposition, analogous to that achieved by POD, facilitates efficient modeling of flow dynamics. The decoder subsequently reconstructs the spatiotemporal evolution from the encoded representation.

To evaluate the effectiveness of the POD-DL model against these fully deep learning models, this chapter investigates three autoregressive strategies for forecasting spatiotemporal flow dynamics using a limited dataset, with the aim of reducing computational costs compared to numerically solving the governing equations. The selected models, include the hybrid POD-DL, presented in Chapter 2, alongside with two deep learning–only approaches grounded in the contemporary literature about spatiotemporal forecasting. These are: (1) a residual

autoencoder (RAE), selected for its proven capacity to model spatiotemporal patterns by coupling a convolutional autoencoder for spatial encoding with a convolutional long short-term memory (ConvLSTM, Shi et al., 2015) network for temporal prediction, and (2) a variational autoencoder (VAE), which employs a similar architectural scheme, integrating a convolutional autoencoder with an LSTM network, while embedding the modeling process within a probabilistic framework.

Note, autoencoders rely entirely on neural networks to compress high-dimensional input data into a lower-dimensional latent space, thereby capturing complex, nonlinear spatial relationships. In contrast, POD performs dimensionality reduction through projection onto an orthogonal basis derived from the most energetic modes of the system, providing a linear representation that emphasizes dominant flow structures.

While nonlinear deep learning autoencoders generally demonstrate enhanced compression capabilities compared to POD, they often lack interpretability in relation to the underlying physical phenomena (Eivazi et al., 2022; Fukami, Fukagata, and Taira, 2020; Fukami, Nakamura, and Fukagata, 2020; Muñoz et al., 2023; Murata et al., 2020). A notable exception is the linear autoencoder, whose latent representations have been shown to closely resemble POD modes (Milano and Koumoutsakos, 2002). For clarity, the term autoencoder will henceforth refer exclusively to nonlinear deep learning autoencoders, unless otherwise specified.

In this context, the objective of this chapter is to shown the effect that these architectures have in the final flow dynamics predictions. In terms of number of training samples required, training time, predictions accuracy and model’s flexibility. In this sense, three datasets representing a flow past a circular cylinder at laminar and turbulent regime, and a isothermal subsonic jet at turbulent regime.

This chapter is organized as follows: Section 3.1 introduces the three models under investigation, POD-DL, the residual autoencoder (RAE), and the variational autoencoder (VAE). Section 3.2 details the training procedures and presents the results for each model across the evaluated datasets. Finally, Section 3.3 provides a critical discussion of the obtained results.

3.1 Methodology

Time series forecasting aims to generate accurate predictions of future states, $\{\mathbf{v}_{t+1}, \mathbf{v}_{t+2}, \dots\}$, based on a sequence of past observations, $\{\mathbf{v}_1, \dots, \mathbf{v}_t\}$. In this chapter, three distinct methodologies are examined and compared: two that are purely based on deep learning (DL), and the hybrid POD-DL model presented in Chapter 2.

Forecasting methodologies available in literature, and studied in this chapter, are classified into two primary categories (Januschowski et al., 2020): point forecasting and probabilistic forecasting (Gneiting and Katzfuss, 2014). Here, point forecasting focuses on approximating a deterministic function f , which may be defined as follows,

$$f : [C \times N_x \times N_y \times N_z \times m] \rightarrow [C \times N_x \times N_y \times N_z \times 1]. \quad (3.1)$$

Similar as before, C denotes the component dimension of the dataset, encompassing variations such as different velocity components, Reynolds numbers, and initial conditions. The indices N_x , N_y and N_z correspond to the spatial discretization along the x -, y - and z -axes, respectively. The parameter m specifies the length of the historical time window used as input for prediction, while the output dimension of 1 indicates a one-step-ahead forecast. Multi-step predictions, as discussed in Section 3.2, are obtained via an autoregressive procedure. Further details on the dataset structure and preprocessing methodology are provided in Appendix A. More precisely this function f represents the forecasting process, as follows,

$$f(\{\mathbf{v}_{s-m+1}, \dots, \mathbf{v}_s\}) = \hat{\mathbf{v}}_{s+1} \simeq \mathbf{v}_{s+1}. \quad (3.2)$$

In this work, the POD-DL model and a residual autoencoder are utilized to approximate this function. The loss function used to fit these models is the MSE, defined as in eq. (2.24).

On the other hand, the probabilistic forecasting framework closely resembles point forecasting, with the critical distinction that, rather than approximating a deterministic function, it seeks to approximate the underlying probability distribution governing the forecasted variables. To this end, a variational autoencoder is employed (D. P. Kingma and Welling, 2013). The following subsections provides a detailed description of the proposed models, beginning with the two models situated within the point forecasting framework.

3.1.1 Point forecasting: POD-DL and residual convolutional autoencoder (RAE)

The aim of point forecasting methodology is to generate deterministic predictions by approximating a function, as defined in eq. (3.1). The models adopt dimensionality reduction strategies, either through an encoder in the case of the RAE, or via POD in the POD-DL framework. Within this reduced-dimensional latent space, DL architectures are employed to forecast the temporal evolution of the flow. To reconstruct the original high-dimensional representation, the RAE utilizes a decoder, while the POD-DL model applies the inverse POD transformation.

As a brief reminder of the POD-DL model, SVD (Sirovich, 1987) is applied to decompose a high-dimensional dataset $\tilde{\mathbf{D}}$ in a product of matrices, as shown in eq. (2.10).

In most applications, it is sufficient to retain only the most energetic POD modes, that is, the leading columns of the matrix $\mathbf{U}$, which correspond to the dominant coherent structures in the flow dynamics (Holmes et al., 1996). This mode truncation is typically guided by examining the singular values (i.e., the diagonal entries of the matrix $\mathbf{\Sigma}$), which quantify the relative contribution of each mode to the overall flow energy. The temporal evolution of these modes is described by the POD coefficients, represented by the rows of matrix $\mathbf{\Gamma}$.

Forecasting is subsequently performed in the space of POD coefficients, where their temporal evolution is predicted using deep learning architectures, an approach conceptually analogous to Galerkin projection methods, as discussed in Chapter 2.

This encapsulates the primary advantages of the POD-DL model over the other approaches

considered: (1) truncation of the POD modes allows the forecasting to be concentrated on the most dynamically significant flow scales, omitting smaller-scale structures that are often uncorrelated and not essential for the target application, (2) the use of SVD provides a computationally efficient means of extracting the POD modes, and (3) by transforming a high-dimensional tensor into a matrix representation (i.e., Σ), where columns correspond to the temporal evolution, POD reduces the complexity of the forecasting model architecture, thereby mitigating the risk of overfitting and enhancing generalization performance.

Similar to Chapter 2, the temporal evolution of the POD coefficients is modeled using a long short-term memory (LSTM) network. Previous studies, such as Parish and Carlberg, 2020, have demonstrated that LSTM architectures outperform traditional forecasting methods in terms of both accuracy and computational efficiency. This advantage stems from the fact that conventional approaches often demand substantial computational resources and exhibit limited generalization to unseen or varying conditions. Similar trends were observed in the M4 forecasting competition, where machine learning models consistently outperformed classical probabilistic forecasting methods (Makridakis et al., 2018). A visual depiction of the POD-DL ROM is shown in Fig. 2.1.

The second model, inspired by prior studies (Han et al., 2019; Hasegawa et al., 2020; Nakamura et al., 2021), employs a residual convolutional autoencoder (RAE), as illustrated in Fig. 3.2. The detailed architecture of this model, as implemented for one of the datasets, is provided in Tabs. C.2, C.3, and C.4 in Appendix C.

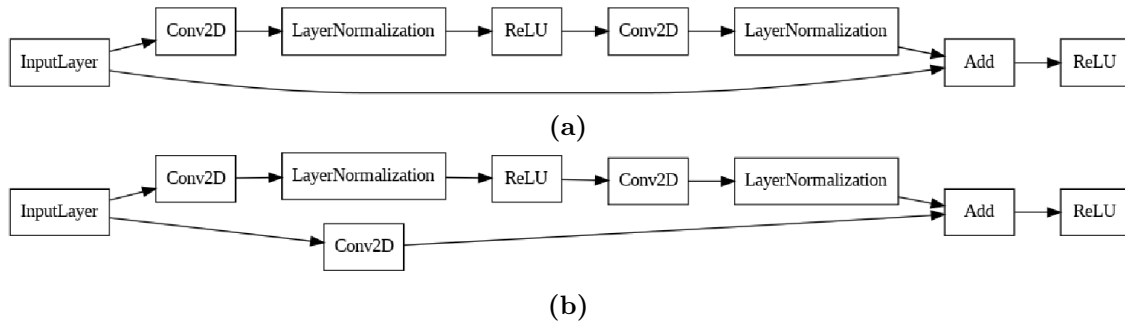


Figure 3.1: Visual representation of the two residual block types used in the RAE: identity block (top) and convolutional block (bottom). Both structures implement residual connections by summing the block’s input with its output. The key distinction lies in their effect on spatial dimensions, the identity block preserves the input resolution, whereas the convolutional block reduces it through downsampling.

This model is defined by an encoder-decoder structure, where the encoder is composed of residual convolutional networks (ResCNNs) (He et al., 2016), wherein the input and output of a convolutional neural network (CNN) layer are combined through element-wise addition. These ResCNNs are structured using two fundamental components: identity blocks and convolutional blocks, as illustrated in Fig. 3.1. While both blocks implement residual connections, the identity block preserves the spatial dimensions and number of channels of the input, whereas the convolutional block modifies them through downsampling or channel expansion.

This architecture improves training efficiency and model robustness, leveraging the principle that it is computationally simpler for a residual block to learn the identity mapping (achieved by setting convolutional weights to zero) than for a standard CNN layer. In contrast, the decoder comprises transposed convolutional layers, which invert the downsampling process of the encoder, enabling reconstruction from the latent space back to the high-dimensional representation, as depicted in Fig. 3.2.

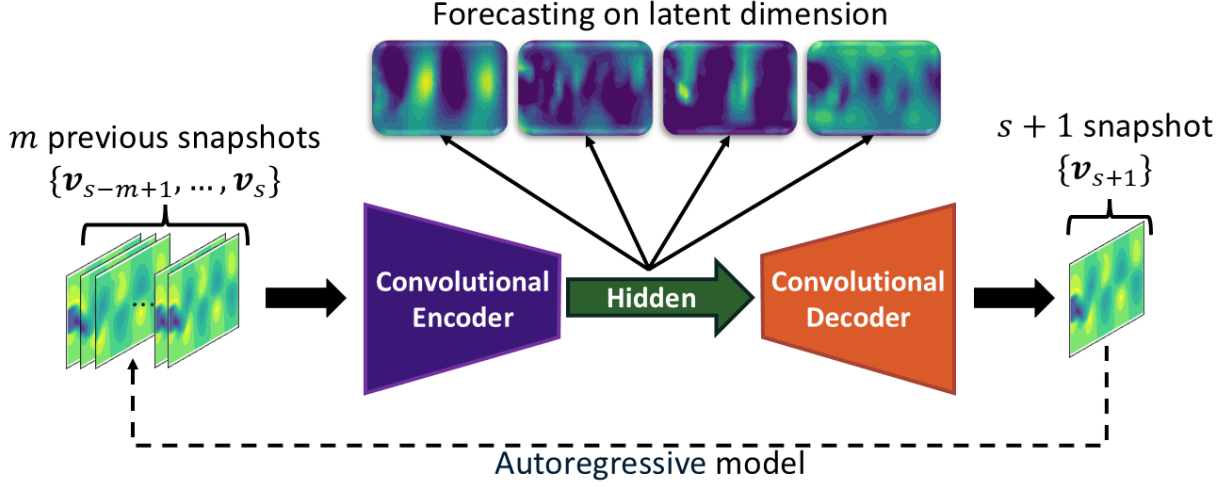


Figure 3.2: Overview of the residual autoencoder (RAE) architecture. Note, in contrast to the POD-DL model, both the encoder and decoder components of this architecture are constructed entirely from CNNs.

It is important to note that both the encoder and decoder operate exclusively on the spatial dimensions of the snapshots, leaving the temporal dimension unaltered. Representative outputs from the encoder, generated using snapshots from the dataset introduced in Chapter 2, which describes the velocity field of a laminar flow past a circular cylinder (Le Clainche et al., 2018), are presented in Fig. C.1 in Appendix C. In contrast to the latent variables derived from POD, which are hierarchically ordered based on their associated cumulative energy, the latent variables produced by autoencoders lack an inherent ordering. Nevertheless, various methods have been proposed to impose a meaningful ranking on these variables, thereby identifying those that contribute most significantly to the underlying dynamics (Fukami, Nakamura, and Fukagata, 2020; Muñoz et al., 2023).

To capture temporal correlations within the input sequence of snapshots $\{v_{s-m+1}, \dots, v_s\}$, a hidden layer comprising a convolutional long short-term memory (ConvLSTM, Shi et al., 2015) unit is employed. As shown in Fig. 3.2, this layer is positioned between the encoder and decoder and operates on the latent variables, which encode the dominant spatial features or patterns of the flow. By modeling temporal evolution in this reduced representation, the ConvLSTM facilitates effective forecasting of the flow dynamics. The detailed architectural specifications of the hidden layer are provided in Tab. C.3.

For both the residual and variational autoencoders, a convolutional LSTM (ConvLSTM) is employed instead of the standard LSTM used in the POD-DL model. This choice is motivated by the nature of the input data: the autoencoders operate on flow snapshots,

which inherently possess spatial structure. Standard LSTM layers require vectorized inputs, necessitating the flattening of latent variables produced by the encoder. Such flattening would not only discard spatial correlations but also significantly increase the number of trainable parameters. In contrast, ConvLSTM layers accept matrix-formatted inputs, thereby preserving spatial information and enabling a more natural and efficient representation of spatiotemporal dynamics.

This problem is not encountered in the POD-DL model because the entries to this model are vectors (the POD coefficients), allowing to use the LSTM architecture.

3.1.2 Probabilistic forecasting: Variational autoencoder (VAE)

In contrast to point forecasting, the objective here is to approximate the probability distribution that governs the transition from a sequence of prior snapshots $\{\mathbf{v}_{s-m+1}, \dots, \mathbf{v}_s\}$ to the subsequent snapshot $\{\mathbf{v}_{s+1}\}$. To this end, we introduce a third model that adopts a Bayesian perspective on statistical inference, implementing a variational inference framework (Blei et al., 2017) based on variational autoencoders (VAEs, D. P. Kingma and Welling, 2013). A visual representation of the proposed architecture is provided in Fig. 3.3.

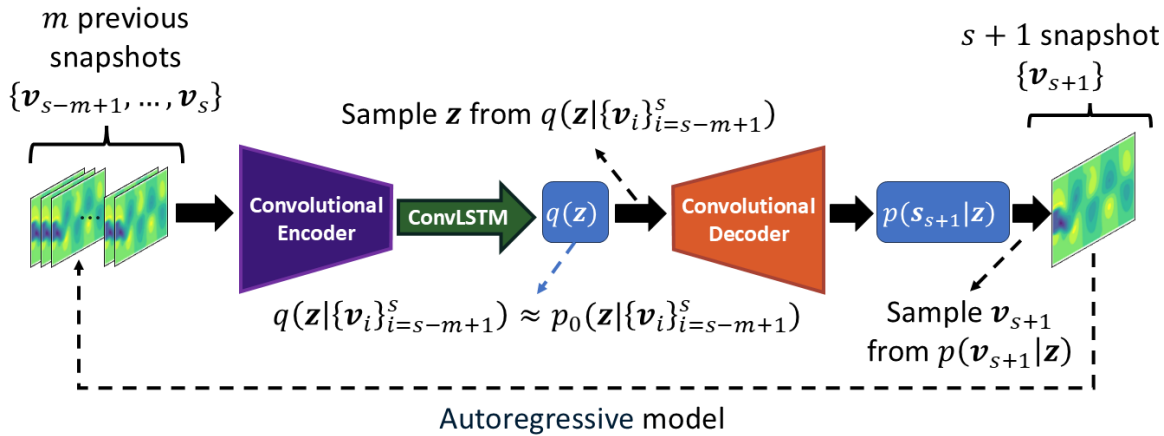


Figure 3.3: Overview of the VAE architecture. Similar to the residual autoencoder, the encoder and decoder of this model are composed of CNNs.

Similar to the residual autoencoder, VAE adopts an encoder–decoder architecture that operates on the spatial dimensions of the input snapshots. Additionally, a ConvLSTM layer is incorporated to capture temporal correlations within the latent space, thereby enabling the modeling of spatiotemporal dynamics. As with RAE the architecture of VAE is dependent on the dataset’s spatial resolution. The specific architecture implemented for the flow past a cylinder at laminar regime is detailed in Tab. C.5 and Tab. C.6.

In contrast to both RAE and the POD-DL model, VAE seeks to approximate the conditional posterior probability function $p_0(\mathbf{z}|\{\mathbf{z}_i\}_{i=s-m+1}^s)$ of the latent variables $\mathbf{z}$, given the input sequence of snapshots $\{\mathbf{v}_{s-m+1}, \dots, \mathbf{v}_s\}$.

This conditional posterior $p_0(\cdot)$ is assumed to be parametric, with the encoder and ConvLSTM layer providing the necessary information to infer its optimal parameters. Furthermore,

the likelihood $p(\mathbf{v}_{s+1}|\mathbf{z})$ of the future snapshot $\{\mathbf{v}_{s+1}\}$ is approximated based on the latent variables $\mathbf{z}$.

As illustrated in Fig. 3.3, the input to the decoder is drawn from the posterior distribution $p_0(\mathbf{z}|\{\mathbf{v}_i\}_{i=s-m+1}^s)$. Consequently, it is necessary to sample from this posterior distribution to provide a latent representation for the decoder. Similarly, the likelihood $p(\mathbf{v}_{s+1}|\mathbf{z})$ is utilized to generate the predicted future snapshot, based on the sampled latent variables.

The original VAE was introduced by D. P. Kingma and Welling, 2013 and it was designed to approximate the probability distribution underlying a collection of images. Once trained, the model allows for the generation of new images by sampling from the approximated posterior distribution, passing the sample through the decoder, and reconstructing an image from the likelihood. In this framework, the VAE effectively functions as a probabilistic image generator. The training objective is based on maximizing the Evidence Lower Bound (ELBO), which is defined as follows,

$$\text{ELBO} = \mathbb{E}[\log p(\mathbf{x}|\mathbf{z})] - \text{KL}[q(\mathbf{z}|\mathbf{x})||p_0(\mathbf{z}|\mathbf{x})]. \quad (3.3)$$

Here $\mathbf{x}$ denotes the input image, $p_0(\mathbf{z}|\mathbf{x})$ represents the true posterior distribution of the latent variables $\mathbf{z}$, $q(\mathbf{z}|\mathbf{x})$ is the variational approximation to this posterior, and $p(\mathbf{x}|\mathbf{z})$ denotes the likelihood of the image given the latent variables. The term $\text{KL}[P||Q]$ refers to the Kullback–Leibler divergence, which quantifies the difference between two distributions P and Q , and it is defined as follows,

$$\text{KL}(P||Q) = \sum_{\mathbf{x} \in \mathcal{X}} P(\mathbf{x}) \log \left(\frac{P(\mathbf{x})}{Q(\mathbf{x})} \right). \quad (3.4)$$

It is important to note that in the original ELBO formulation given in Equation (3.3), the bound is evaluated for a single image $\mathbf{x}$.

In this chapter, a modification of the original loss function (3.3) is explored, wherein both the exact and approximate posterior distributions are conditioned on a sequence of input snapshots $\{\mathbf{v}_i\}_{i=s-m+1}^s$, rather than on a single image $\mathbf{x}$. Additionally, we aim to approximate the likelihood of the time-ahead snapshot $p(\mathbf{v}_{s+1}|\mathbf{z})$. Accordingly, the loss function used to train the VAE in this spatiotemporal forecasting framework is defined as follows,

$$\begin{aligned} \text{ELBO}_f = & \mathbb{E}[\log p(\mathbf{v}_{t+1}|\mathbf{z})] - \\ & \text{KL}[q(\mathbf{z}|\{\mathbf{v}_i\}_{i=s-m+1}^t)||p_0(\mathbf{z}|\{\mathbf{v}_i\}_{i=s-m+1}^s)]. \end{aligned} \quad (3.5)$$

This modified loss function enables the approximation of the likelihood of the future snapshot based on latent variables $\mathbf{z}$, which are inferred from the preceding sequence of snapshots $\{\mathbf{v}_i\}_{i=s-m+1}^s$. As the primary objective of this thesis is forecasting, the training of the VAE is guided by the forecasting-specific evidence lower bound, denoted as ELBO_f (eq. (3.5)), rather than the standard ELBO formulation given in eq. (3.3).

3.2 Results

The three models introduced in Section 3.1 are evaluated using two primary datasets, each representing a flow past a circular cylinder under laminar and turbulent regimes, respectively, the former being the same dataset employed in Chapter 2. These datasets are used to assess model performance in terms of prediction accuracy with limited training data, computational efficiency during training, and model stability. In addition, a third dataset, representing the velocity field of an isothermal subsonic jet, is employed to evaluate the transfer learning capabilities of the POD-DL model in comparison to the other two approaches. For clarity and conciseness throughout the remainder of this chapter, the dataset corresponding to the laminar flow past a cylinder will be referred to as *laminar*, while the turbulent case will be referred to as *turbulent*.

It is important to note that the models examined in this chapter are autoregressive in nature, that is, they utilize the predicted snapshot $s + 1$, denoted as $\{\mathbf{v}_{s+1}\}$, alongside the preceding m snapshots $\{\mathbf{v}_{s-m+1}, \dots, \mathbf{v}_s\}$, to form a new input sequence for generating the subsequent prediction $\{\mathbf{v}_{s+2}\}$, and so forth as indicated in equations (2.26, 2.28). This recursive process is repeated iteratively. Similar to Chapter 2, for both datasets, the models are tasked with predicting 200 time-ahead snapshots.

Regardless of the dataset, all models are trained using the Adam optimization algorithm (D. P. Kingma and Ba, 2017), with default parameter values: $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$. Further details on the algorithm and its parameterization can be found in D. P. Kingma and Ba, 2017.

Table 3.1: Learning rates used for training each model. Note that the length of the input snapshot sequence varies by dataset for the RAE and the VAE models, whereas it remains fixed for the POD-DL model.

Model	Learning rate (α)	Length input sequence	# predictions generated
POD-DL	10^{-3}	10	6
Residual AE	5×10^{-4}	10 (laminar) / 5 (turbulent)	1
VAE	5×10^{-5}	10 (laminar) / 5 (turbulent)	1

The optimal training parameters for each model were identified through extensive experimentation and fine-tuning. These parameters vary across models to ensure optimal performance in each specific scenario. Notably, the learning rate α differs between models, as detailed in Tab. 3.1. In addition, both the length of the input snapshot sequence and the number of time-ahead predictions vary depending on the model and dataset, as also summarized in Tab. 3.1. Note that, forecasting multiple future snapshots simultaneously increases the complexity of the learning task and typically requires a larger training dataset to ensure stable and accurate predictions.

Extensive experimentation revealed that both RAE and VAE models exhibit superior perfor-

mance when limited to single-step-ahead predictions, as opposed to forecasting multiple future states simultaneously. In contrast, the POD-DL model, owing to its architectural simplicity and reduced input dimensionality, demonstrates the capability to effectively predict over a longer temporal horizon, even when trained on relatively small datasets.

The number of training and testing samples used for each dataset, except the isothermal jet, is summarized in Tab. 3.3. These values correspond to the optimal training set sizes identified for the RAE and VAE models. A detailed analysis of model performance under varying training set sizes is presented in Section 3.2.3, providing insight into the robustness and generalization capabilities of all three models when exposed to different training set sizes.

Table 3.3: Number of snapshots used for both training and testing the models. Note the laminar flow is composed by three-dimensional snapshots, while the turbulent flow is composed by two-dimensional snapshots.

Dataset	Snapshot dimension	Training	Testing	Total
Laminar	3D	299	200	499
Turbulent	2D	2800	1200	4000

Table 3.4 reports the number of training epochs used for each model across all datasets, with the exception of the isothermal jet case. It also includes the average training and inference times required to compute time-ahead predictions. These times were estimated by repeating both the training and prediction processes multiple times and computing the mean values. The training for each model was terminated when the loss function plateaued, indicating convergence to at least a local minimum. As shown in Tab. 3.4, for the VAE model the training was terminated early, in the turbulent dataset, due to a rapid divergence that did not improve even with a larger training length.

Hyperparameter selection for all models was guided by extensive experimentation, with the objective of balancing model complexity and training efficiency. In the case of the RAE, particular attention was given to optimizing the representation of both spatial and temporal dependencies. While reducing the latent dimensionality can simplify the forecasting task and reduce computational cost, it may also compromise the encoder’s ability to capture essential spatial features, thereby degrading the accuracy of dynamic predictions. Consequently, effective forecasting depends critically on the encoder’s capacity to encode spatial structures accurately, necessitating a careful trade-off in selecting the size of the latent space.

For the VAE, hyperparameter tuning followed a similar strategy to that employed for the RAE, with the added objective of balancing the reconstruction loss and the Kullback–Leibler (KL) divergence to ensure effective training of the variational inference framework.

Finally, for the POD-DL model, hyperparameter tuning focused on selecting an appropriate number of POD modes. The goal was to retain the most energetically significant structures in the flow dynamics while maintaining computational tractability. Increasing the number of retained modes extends the training duration and complexity, as the lower-energy modes often exhibit noisier and less predictable dynamics. Thus, truncating to an optimal number of modes was essential for preserving the core dynamical features of the system while balancing

Table 3.4: Training length measured in number of epochs, training time measured in minutes and prediction time of the time-ahead snapshots measured in seconds for each one of the three forecasting models and each dataset, respectively. Note the laminar flow is three-dimensional, and the turbulent flow is two-dimensional. Notably, the VAE model exhibited a markedly reduced training length for the turbulent dataset due to a rapid divergence early in training, after which convergence could not be restored regardless of the number of training epochs.

Dataset	Model	Training length (epochs)	Training time (min.)	Prediction time (sec.)
Laminar				
	POD-DL	420	4.677	2.415
	Residual AE	30	280.623	165.071
	VAE	30	86.269	45.631
Turbulent				
	POD-DL	420	30.418	1.683
	Residual AE	120	329.665	29.094
	VAE	14	13.966	14.746

model accuracy and computational efficiency.

The evaluation metrics employed in this chapter to assess predictive reliability vary depending on the dataset. For the laminar flow, the goal is to enable one-to-one comparisons between the predicted and ground truth dynamics, thus, the two metrics used in Chapter 2 for this dataset: RMSE (2.22) and SSIM (2.23), are also used in this chapter. In contrast, the inherently stochastic nature of the turbulent flow necessitates a shift in focus, from exact trajectory matching to statistical consistency, so percentiles from the predicted and ground truth velocity are compared as well as the histograms.

The following sections present the temporal predictions obtained from the three models when applied to the laminar (Section 3.2.1) and turbulent flows (Section 3.2.2).

3.2.1 Three-dimensional wake behind a circular cylinder: laminar regime

The dataset utilized in this section is the same as the one presented in Chapter 2, derived from a numerical simulation of a three-dimensional wake behind a circular cylinder, a bed problem in fluid dynamics (Le Clainche et al., 2018). In this case the initial 100 snapshots represent the transient phase of the flow, during which statistical properties are continuously evolving, as illustrated in Fig. 3.4. Due to the non-stationary nature of this regime, these initial snapshots are excluded from both training and testing procedures. As shown in Fig. 3.4, the dataset is partitioned such that snapshots from 100 to 399 are used for training, while those from 400 to 599 are reserved for testing. Importantly, these subsets are mutually

exclusive, ensuring that the models are evaluated on data not encountered during training. This setup enables a rigorous assessment of the models' capacity to generalize and accurately forecast long-term temporal dynamics.

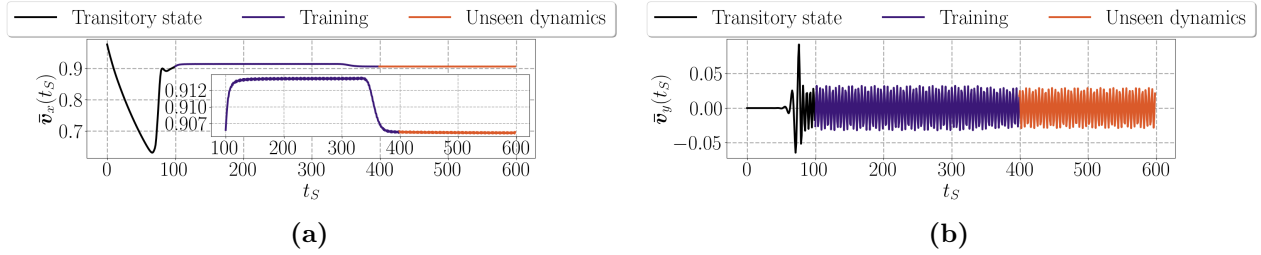


Figure 3.4: (a) Streamwise and (b) wall-normal velocity evolution in the laminar flow. Black, blue and orange lines represent the transitory state of the flow, the portion of data used for training and the portion used for evaluating the models, respectively.

Given the spatially three-dimensional nature of this dataset, the number of training instances can be substantially increased through the data augmentation strategy detailed in Appendix A. This approach expands the training dataset from 299 original samples to $299 \times 64 = 19,136$ samples, where the factor 64 corresponds to the number of discretization points along the z -axis (spanwise direction). It is important to note that this augmentation procedure is applied exclusively to the residual and variational autoencoders. In contrast, the POD-DL framework incorporates POD, as shown in Fig. 2.1, a non-parametric method that projects the high-dimensional snapshots into a low-dimensional space, making such augmentation unnecessary.

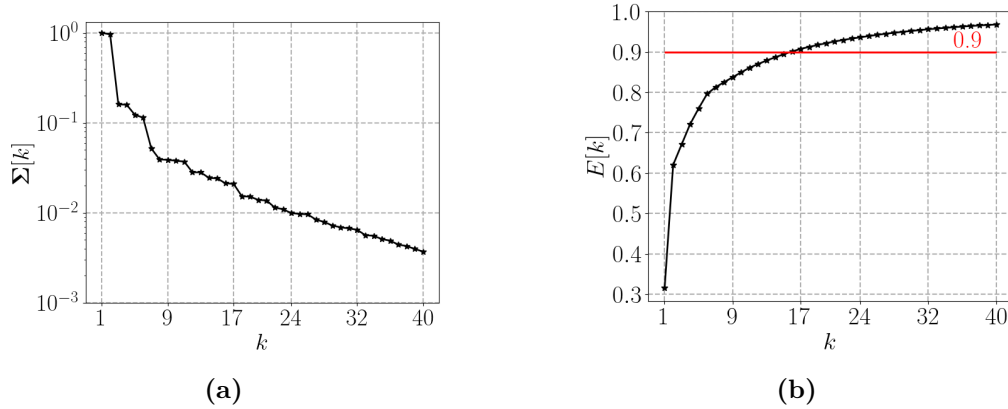


Figure 3.5: (a) Mode decay and (b) cumulative energy for the 40 most energetic modes in the laminar flow.

Figure 3.5 presents the decay profile of the singular values corresponding to the 50 most energetic modes, obtained by applying POD to this dataset, along with the cumulative energy E , which is computed following eq. (2.25). In accordance with Abadía-Heredia et al., 2022, the first 20 modes are selected for forecasting the dynamics of this laminar flow. As shown in Fig. 3.5, these number of modes collectively capture over 90% of the total energy, which is desirable for a flow at laminar regimen.

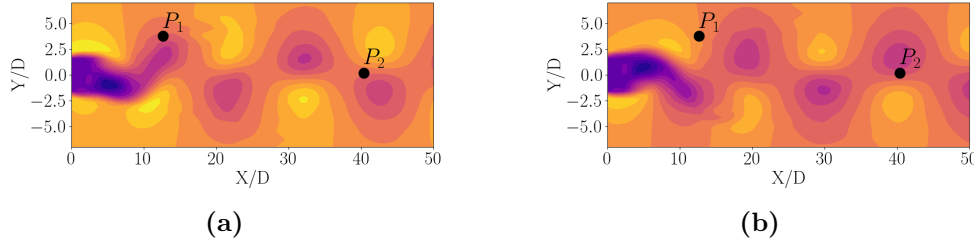


Figure 3.6: Snapshots of the streamwise velocity field from the laminar flow at time instants (a) 450 and (b) 485, highlighting in red points the spatial locations P_1 (left) and P_2 (right). These reference points are used to evaluate the temporal evolution of the streamwise (v_x) and wall-normal (v_y) velocity components, as presented in Fig. 3.7 and Fig. 3.8.

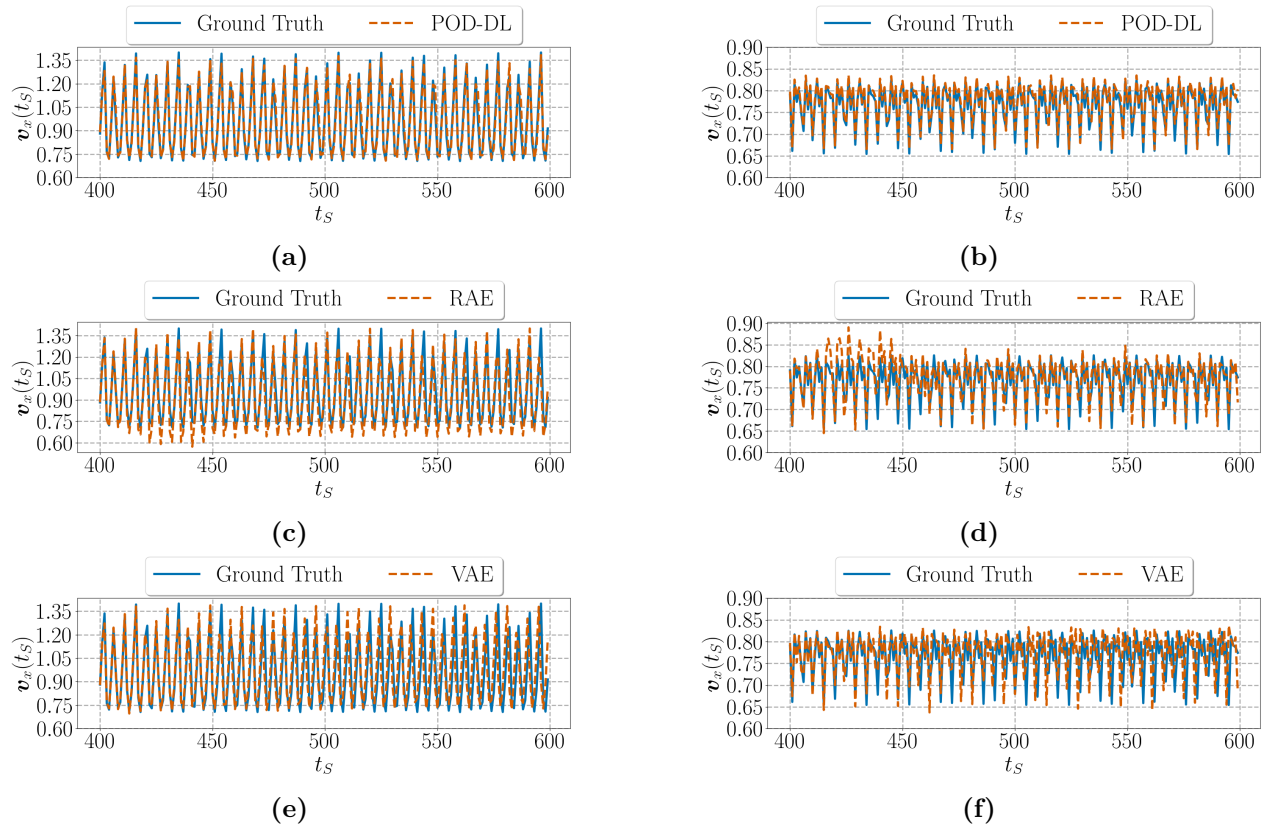


Figure 3.7: Comparison of the ground truth streamwise velocity evolution $u_x(t)$, at spatial coordinate P_1 shown in Fig. 3.6, against the predictions obtained from the (a) POD-DL, (b) residual autoencoder and (c) variational autoencoder models, respectively. Counter part for the spatial coordinate P_2 in figures (d), (e) and (f). In all figures the ground truth is represented by a solid line and the prediction by a dashed line.

Once computed the POD decomposition, via SVD as in eq. (2.10), and following the training phase, each model is tasked to perform autoregressive predictions of 200 future snapshots, spanning from time step 400 to 599, which are then compared against the corresponding ground truth data. This evaluation is first qualitatively conducted by comparing the velocity evolution at two representative spatial coordinates depicted in Fig. 3.6. The comparison of

the real streamwise velocity, against predictions from the POD-DL, RAE and VAE models, is presented in Fig. 3.7, while Fig. 3.8 shows analogous results for the wall-normal velocity.

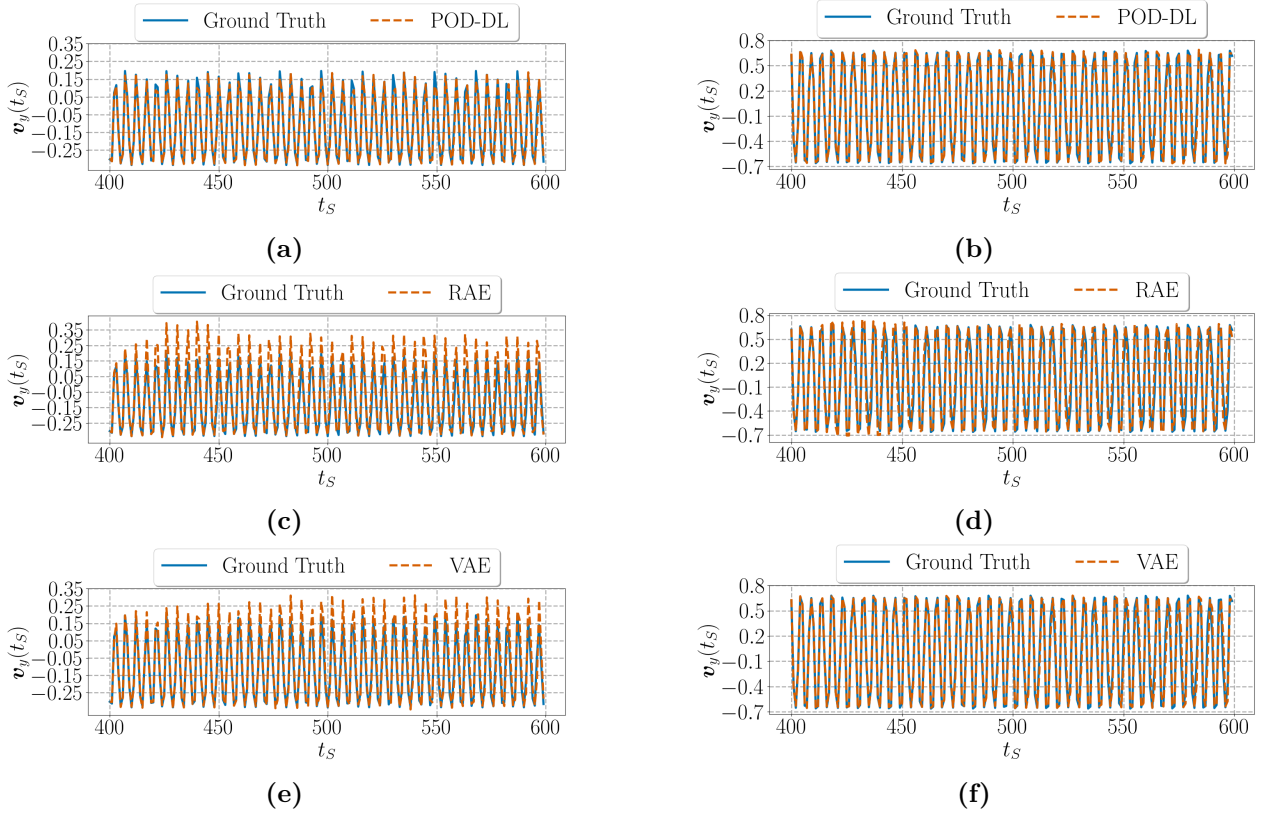


Figure 3.8: Same as Fig. 3.7 for the wall-normal velocity, $v_y(t_s)$.

As illustrated in Fig. 3.7, the POD-DL model yields the closest predictions to the ground truth for the streamwise velocity. A comparable trend is observed in Fig. 3.8 for the wall-normal velocity, where the POD-DL again demonstrates superior predictive accuracy. While the RAE outperforms the VAE in the streamwise component, it yields worse predictions for the wall-normal component. In either cases, both the RAE and VAE are outperformed by the POD-DL.

To quantitatively measure the predictive accuracy of the models, both the RMSE and the SSIM are used, which are defined in eq. (2.22) and eq. (2.23). Recall the SSIM evaluates the perceptual similarity between images by comparing structural information. The performance of each model on the laminar flow dataset, as measured by these metrics, is presented in Fig. 3.9.

Consistent with the qualitative comparisons shown in Fig. 3.7 and Fig. 3.8, the POD-DL model achieves the highest accuracy according to both the RMSE and the SSIM. Given the autoregressive nature of the forecasting, it is expected that prediction errors accumulate over time, degrading the quality of long-term predictions. Nonetheless, the POD-DL model demonstrates enhanced temporal stability, with significantly lower error propagation over extended horizons.

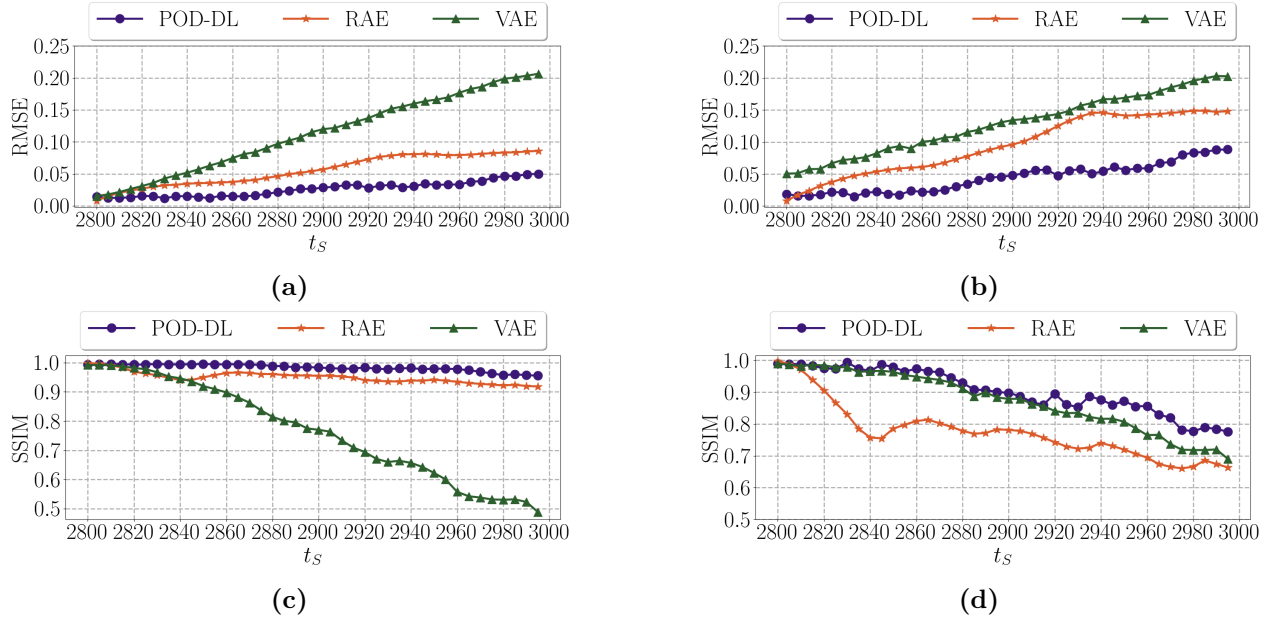


Figure 3.9: Prediction error in the laminar flow scenario as quantified by (a) the RMSE for the streamwise velocity component, (b) the RMSE for the wall-normal component, (c) the SSIM for the streamwise component and (d) the SSIM for the wall-normal component. These metrics evaluate the performance of each model in capturing the temporal dynamics and structural fidelity of the velocity fields.

This relative stability is also qualitatively evident in Fig. 3.10 and Fig. 3.11, which compare representative snapshots corresponding to the ground truth and predicted dynamics from all three models. Note, at time step 550 in both figures, either the RAE and VAE predictions exhibit spatial displacements, whereas the POD-DL predictions remain more closely aligned with the reference, despite minor deviations.

Based on these findings, it can be concluded that the POD-DL model most effectively captures the underlying flow dynamics, followed by the RAE, while the VAE yields the least accurate predictions. The observed disparity in accuracy may be attributed to the complexity of the ELBO loss function, defined in eq. (3.5), which seeks to approximate an underlying probability distribution. In contrast, both the POD-DL and RAE models are trained using the comparatively simpler MSE loss, defined in eq. (2.24). The increased complexity of the ELBO may necessitate a larger volume of training data for the VAE to achieve a predictive performance similar to the other models.

3.2.2 Three-dimensional wake behind a circular cylinder: turbulent regime

The second test case to evaluate the three models is a time-resolved PIV dataset of the three-dimensional turbulent wake of a circular cylinder (Mendez et al., 2020), which is obtained from experimental measurements allowing to test the models in a real case scenario. As stated at the beginning of Section 3.2, this dataset is referred to as the *turbulent* flow for simplicity.

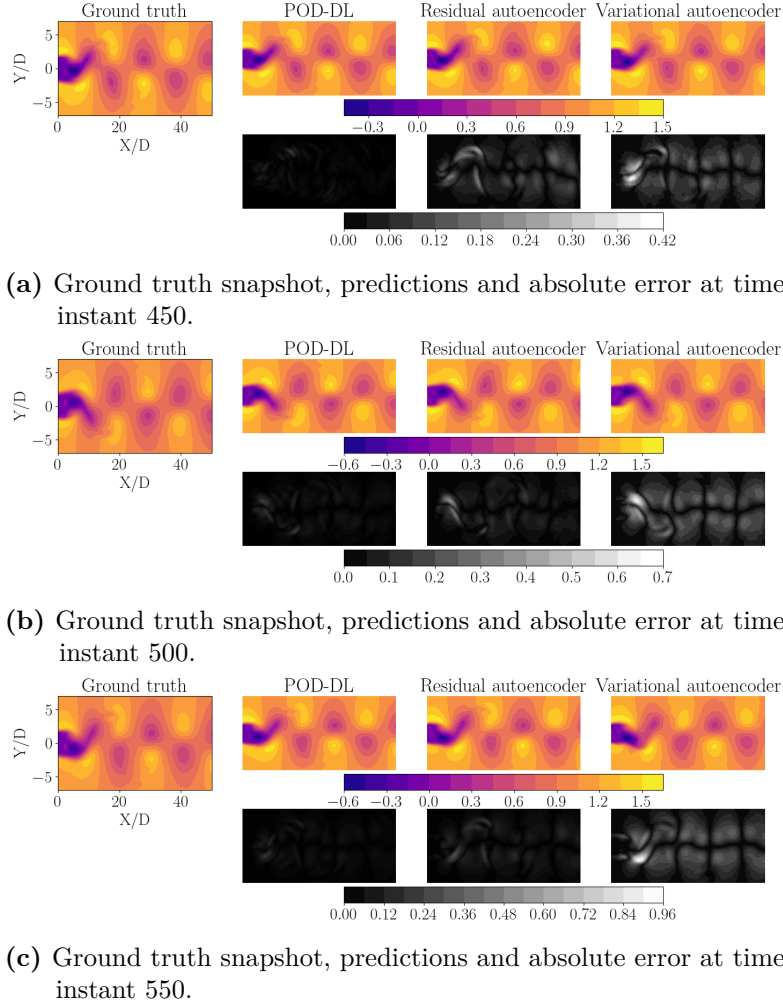


Figure 3.10: Representative snapshots of the streamwise velocity component from the laminar flow at time instants 450, 500, and 550, displaying the ground truth, model predictions and corresponding absolute error maps in grayscale. Results are shown for each of the evaluated models to highlight predictive accuracy and spatial error distribution over time.

All experiments were carried out in the L10 low-speed wind tunnel at the von Karman Institute (cross-section $20 \text{ cm} \times 20 \text{ cm}$) with a circular cylinder of diameter $D = 5 \text{ mm}$ and span $L = 200 \text{ mm}$. The experimental model was a smooth aluminium rod mounted horizontally between end plates, so that the measurement plane (streamwise x , wall-normal y) passed through the mid-spanwise at $z = 0$.

Two free-stream velocities were programmed:

- High-speed phase: $U_\infty = 12.1 \text{ m/s}$, $\text{Re} \simeq 4.0 \times 10^3$,
- Low-speed phase: $U_\infty = 7.9 \text{ m/s}$, $\text{Re} \simeq 2.6 \times 10^3$.

The velocity was changed smoothly between these values over ~ 1 second, so the vortex-shedding frequency changed from 450 to 303 Hz while the Strouhal number (St) remained

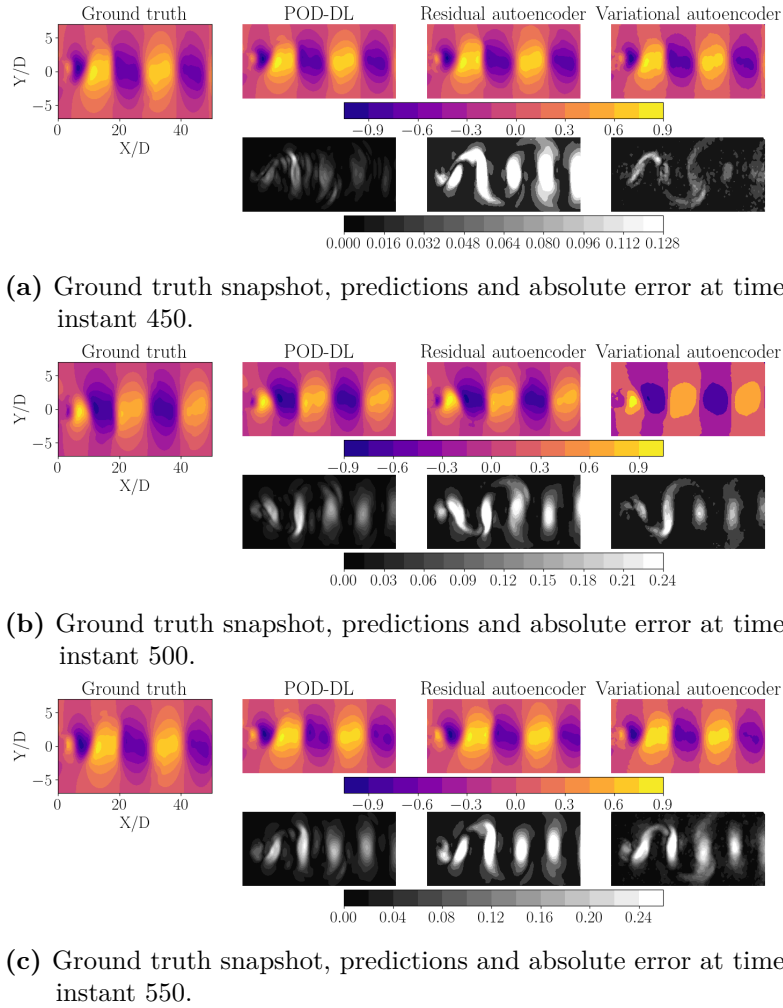


Figure 3.11: Representative snapshots of the wall-normal velocity component from the laminar flow at time instants 450, 500, and 550, displaying the ground truth, model predictions and corresponding absolute error maps in grayscale. Results are shown for each of the evaluated models to highlight predictive accuracy and spatial error distribution over time.

nearly constant at $St \approx 0.19$.

As specified in Mendez et al., 2020, the flow field was recorded with a double-frame time-resolved PIV by seeding the flow with mineral-oil droplets of mean diameter $\approx 1.5 \mu\text{m}$ and illuminating the measurement plane with a Photonics Industries DM20-527 Nd:YLF laser that delivered 20 mJ per pulse at $f_s = 1 \text{ kHz}$. The images were captured by an Optronis SpeedSense 9090 CMOS camera, whose sensor is cropped to 1280×488 pixels (px), in double-frame mode at a sampling frequency of $f_s = 3 \text{ kHz}$. A $70 \times 26 \text{ mm}$ field of view was projected onto 1280×488 px, giving a spatial resolution of 18.3 px/mm .

The velocity vectors were obtained using the DYNAMICSTUDIO software, reducing the interrogation window from $96 \times 96 \text{ px}$ to $24 \times 24 \text{ px}$ with 50% overlap by calculating the vectors every 12 px. Finally a universal outlier detection with a 3×3 kernel followed by local interpolation

was applied to validate the vector fields.

The camera recorded $S \approx 13500$ double frames over 4.5 seconds, at $f_s = 3.5$ kHz sample frequency, this configuration yields a temporal spacing of 0.333 ms between frames and covers the entire transient experiment.

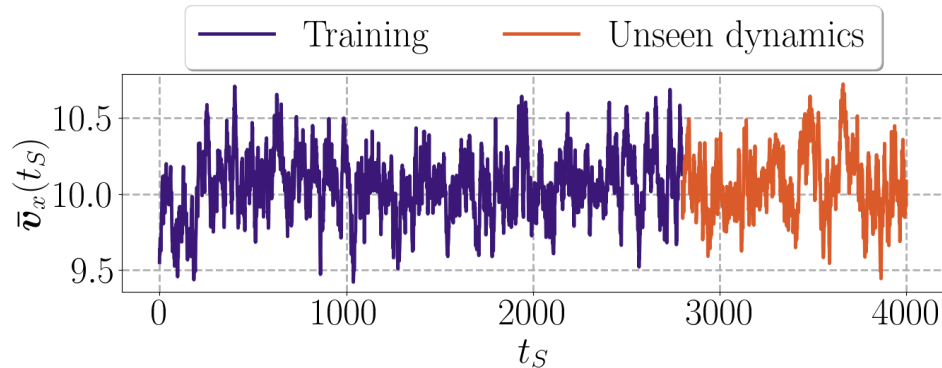


Figure 3.12: Temporal evolution of the spatially averaged streamwise velocity, $\bar{v}_x$, in the turbulent flow. The blue line indicates the portion of the data used for model training, while the orange line represents the unseen dynamics from where the first 200 snapshots are used for evaluating the models.

From this time-span the first 1.33 seconds of the record, i.e., the initial 4000 snapshots, correspond to a stationary wake produced with a free-stream velocity of 12.1 m/s ($\text{Re} \approx 4000$), note during this interval the vortex shedding frequency stays close to 450 Hz. Immediately afterwards the tunnel speed is lowered smoothly over about 1 second, so the wake passes through a controlled transient in which the shedding frequency slides toward 303 Hz while the Strouhal number remains near $\text{St} = 0.19$. The frames acquired after 1.33 seconds therefore contain both the velocity-ramp segment and the subsequent low-speed steady state obtained at 7.9 m/s ($\text{Re} \approx 2600$).

In this chapter only the first 4000 snapshots, representing the steady phase at $\text{Re} = 4000$, are utilised to evaluate the DL models, in Chapter 4 the snapshots corresponding to the steady phase at $\text{Re} = 2600$ are utilized as test case while in Chapter 5 snapshots corresponding to both steady phases are used to train and evaluate the ROM model. However, of these initial 4000 snapshots, the first 2800 are allocated for training, while the subsequent 200 snapshots are reserved for testing as shown in Fig. 3.12.

Figure 3.12 illustrates the evolution, in time, of the spatially averaged streamwise velocity component, $\bar{v}_x$. It is evident from the figure that this flow exhibits significantly greater complexity compared to the laminar flow case shown in Fig. 3.4. This increased complexity arises due to the higher Reynolds number at which the flow is investigated.

Through extensive experimentation, it was found that the most effective strategy for handling this dataset involves simplifying the dynamics using SVD. Specifically, the flow is reconstructed by retaining only the most energetic modes, which capture the dominant features of the system's behavior. Figure 3.13 illustrates the decay of the 100 most energetic modes and their corresponding cumulative energy E , which is defined in eq. (2.25).

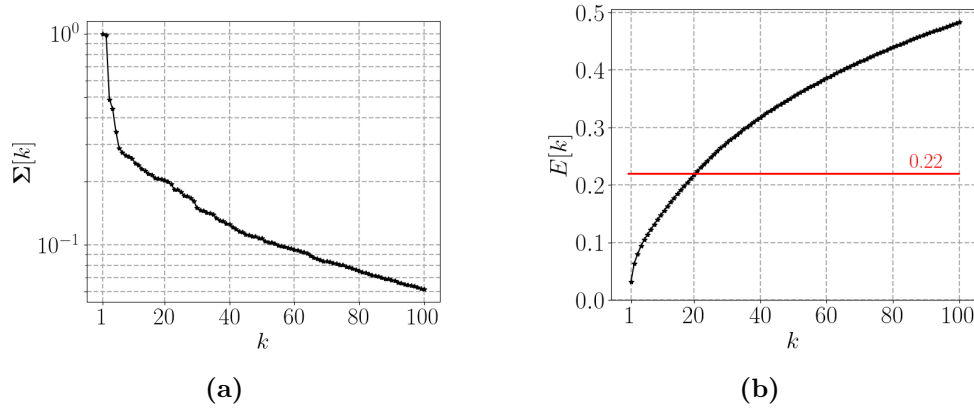


Figure 3.13: (a) Mode decay and (b) cumulative energy for the 100 most energetic modes in the turbulent flow.

It was observed that optimal training performance for both the RAE and VAE models was achieved by retaining the 20 most energetic modes. Figure 3.14 presents a visual comparison between the original flow data, obtained from experimental measurements, and the reconstructed flow field using SVD with these 20 dominant modes. As indicated in Fig. 3.13, this subset of modes captures approximately 22% of the overall flow energy.

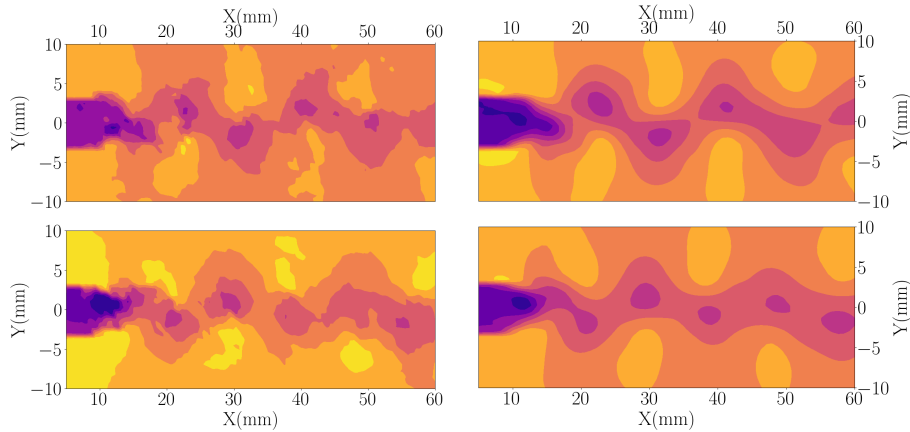


Figure 3.14: Comparison of the original flow from experimental measurements (left column) and the reconstruction using SVD (right column), keeping the 20 most energetic modes to reconstruct the flow field.

This simplified dataset, constructed by retaining only the 20 most energetic modes, is used to train both the RAE and VAE models. In contrast, the POD-DL model, which intrinsically incorporates SVD, is trained on the original dataset. However, for consistency in comparison, the POD-DL model is also restricted to using the same 20 most energetic modes. In all cases, SVD is applied solely to the first 2800 samples, which constitute the training set for all models. As with the laminar flow case, model evaluation is performed on unseen data, specifically, each model is tasked with making autoregressive predictions starting from snapshot 2800 onward.

The predictions computed by each model are subsequently compared against the ground truth data obtained from experimental measurements, in order to evaluate the capability of these reduced-order models to accurately forecast complex flow dynamics.

The rationale for training the models on a simplified dataset is to enhance learning efficiency by concentrating on the principal flow dynamics. This approach aligns with methodologies adopted in previous studies, such as that of Mata et al., 2023.

Following training, the models are tasked with predicting 200 future snapshots, ranging from snapshot 2800 to 2999, and comparing the results with ground truth data obtained from experimental measurements. Similar to the laminar flow, this comparison is first done by examining the predicted and ground truth velocity field at two spatial coordinates identified in Fig. 3.15. The comparison between predicted and ground truth velocity at these two locations is then presented in Fig. 3.16.

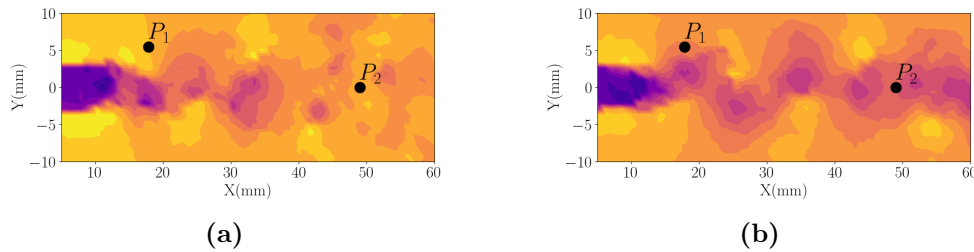


Figure 3.15: Snapshots from the turbulent flow streamwise velocity, at time instants (a) 2800 and (b) 2876, representing by red points, P_1 (left) and P_2 (right), the spatial coordinates used to compare the streamwise velocity evolution shown in Fig. 3.16.

For comparative purposes with the laminar flow case, Fig. 3.17 presents the prediction errors, as measured by RMSE and SSIM, for all three models. As evident from the figure, the predictive accuracy deteriorates significantly, mainly due to the presence of a more complex flow dynamics. Notably, the RMSE values indicate a substantial rise in error, even for the POD-DL model, suggesting that some forecasts may be highly inaccurate. In contrast, the SSIM results provide another perspective: while the VAE exhibits a marked divergence from the reference dynamics, the POD-DL and RAE models still capture key structural features of the flow, indicating partial preservation of physical coherence in their predictions.

The findings presented in Fig. 3.17 are further corroborated by the visual comparison of predicted and actual snapshots at selected time steps, as shown in Fig. 3.18. This comparison clearly demonstrates that both the POD-DL and RAE models are able to reconstruct the dominant flow structures, which are preserved through the truncated POD modes derived from SVD. In contrast, the VAE fails to replicate the true dynamics, displaying a complete departure from the reference dynamics.

The elevated RMSE values observed in Fig. 3.17 can be attributed to the increased complexity of the turbulent flow dynamics in this dataset, making direct snapshot-by-snapshot comparisons between predicted and reference fields less informative than in the laminar case.

In turbulent regimes, as considered in this section, the primary objective often shifts from achieving precise point-wise predictions to accurately reproducing statistical properties of the

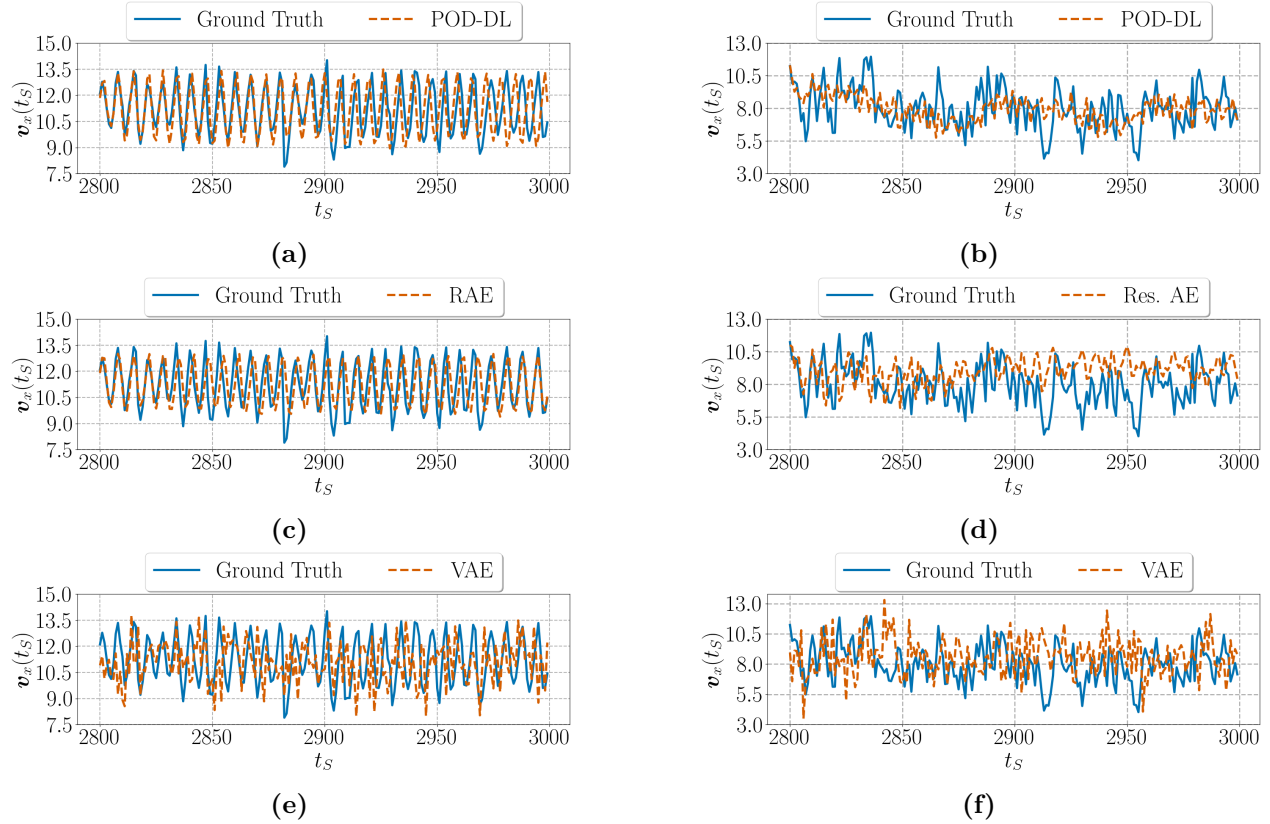


Figure 3.16: Comparison of the ground truth turbulent flow streamwise velocity evolution at the spatial coordinate P_1 shown in Fig. 3.15, against the predictions obtained from the (a) POD-DL, (c) RAE and (e) VAE, respectively. Counter part for the spatial coordinate P_2 in figures (b), (d) and (f). In all figures the ground truth is represented by a solid line and the prediction by a dashed line.



Figure 3.17: Error committed by the models, in each prediction for the streamwise velocity component, measured by the (a) RRMSE and (b) the SSIM.

flow. Therefore, to check how well each model is approximating the underlying distribution of the dynamics, Fig. 3.19 compares the ground truth and predicted probability density function measured by kernel density estimation (KDE). All these metrics further reveals that the POD-DL model yields more precise and temporally stable predictions compared to the RAE, and particularly outperforms the VAE in terms of predictive consistency and accuracy. This superior performance of the POD-DL model in capturing the essential dynamics and

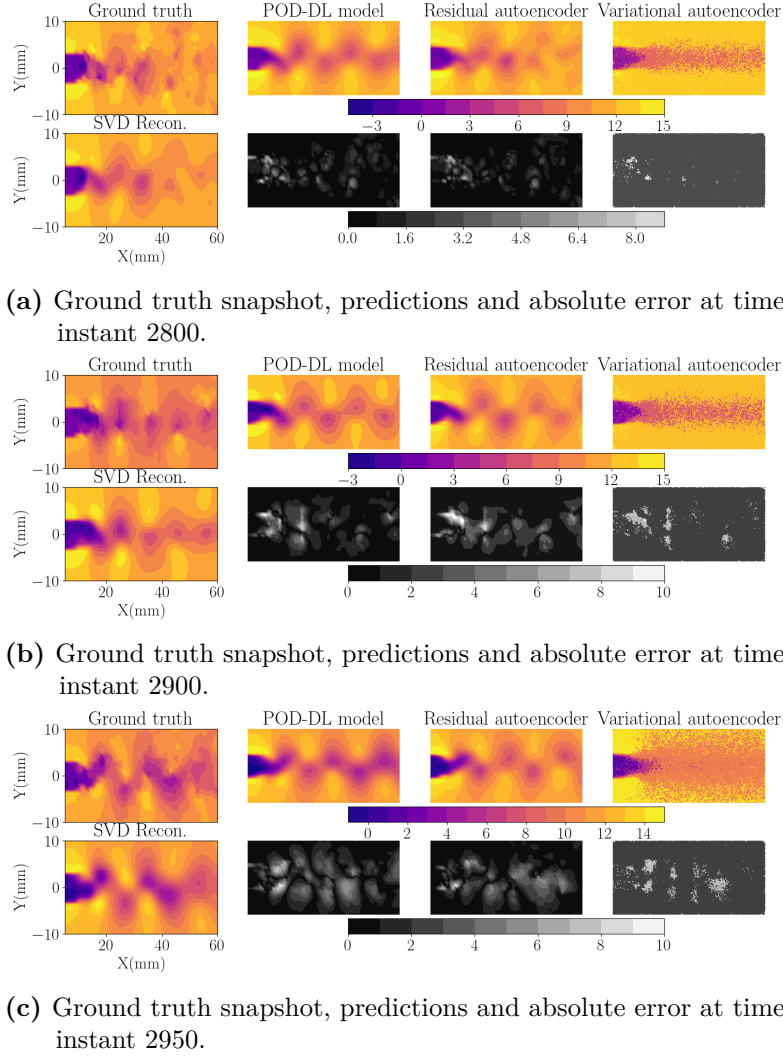


Figure 3.18: Snapshots at some representative time instants showing the ground truth streamwise velocity component of the turbulent flow, the predictions obtained from the different models and their respective absolute error in gray-scale.

delivering accurate predictions underscores its robustness and effectiveness, particularly in the context of complex flows like the ones in turbulent regime.

The results discussed in the preceding sections collectively highlight the superior performance of the POD-DL framework relative to fully deep learning-based models, particularly in terms of predictive accuracy and temporal stability. The subsequent section investigates the robustness of the POD-DL approach when trained on datasets of varying sizes, since so far, all models, including the RAE and VAE, have been trained using the same number of samples.

3.2.3 Stability of POD-DL at different numbers of training samples

This section investigates the training stability of the POD-DL model under reduced training data conditions. The turbulent flow past a circular cylinder (Mendez et al., 2019), characterized

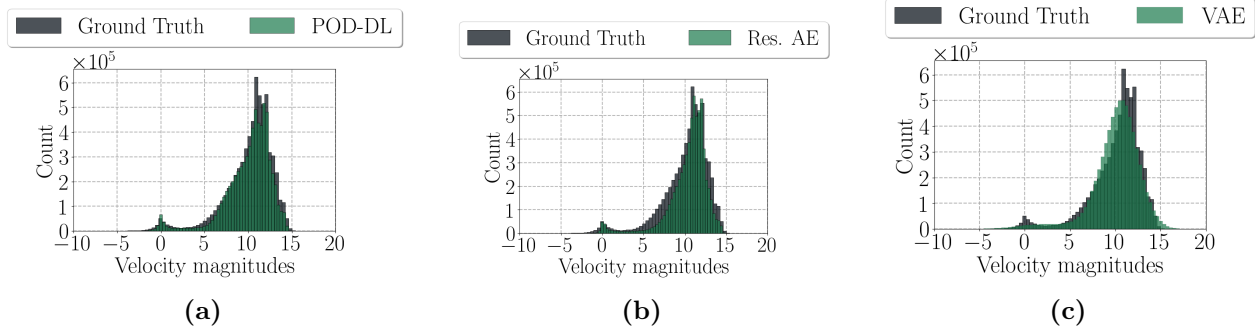


Figure 3.19: Comparison of the probability density function between the ground truth and predicted velocity field of the turbulent flow obtained from (a) POD-DL, (b) RAE and (c) VAE models, respectively.

by its intrinsic complexity, serves as the test case for this analysis.

To this end, the POD-DL model was independently trained using three training sets of differing sizes: 500, 1000, and 2800 samples. The largest of these sets, comprising 2800 samples, corresponds to the training configuration used for the RAE and VAE models, as discussed in Section 3.2.2.

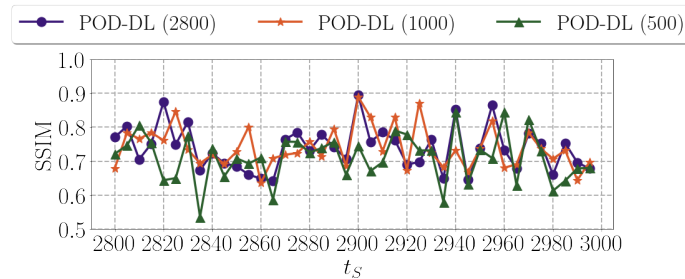
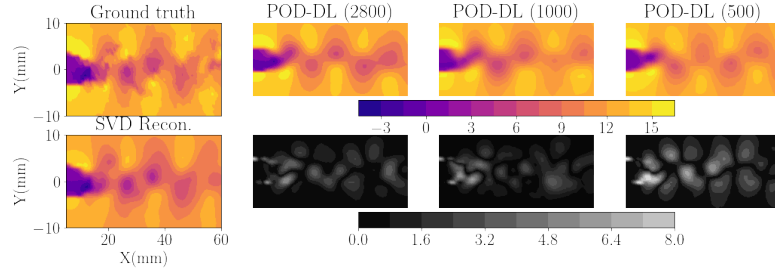


Figure 3.20: Prediction error of the POD-DL model trained with different number of samples (indicated in brackets) for the streamwise velocity component of the turbulent flow past a cylinder. Errors are quantified using (a) the RMSE and (b) the SSIM defined in (2.22) and (2.23), respectively.

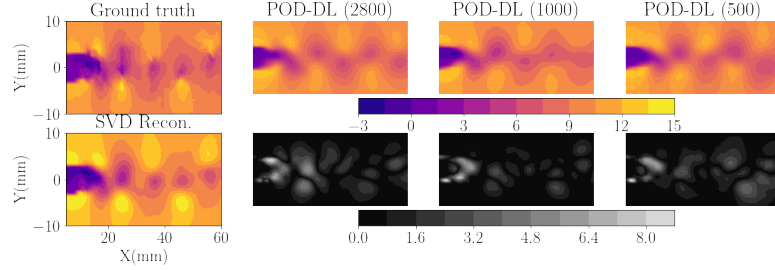
Figure 3.20 presents the prediction error, computed by the SSIM, of the POD-DL model over a horizon of 200 time steps, consistent with the evaluation framework in Section 3.2.2, for each training set size. The results from the SSIM shows a model's robustness to the training set size, with the least accurate predictions corresponding to the case with the smallest training set. This is further observed in Fig. 3.21, which compares predicted and reference snapshots at selected time steps, the use of smaller training sets introduces a slight spatial shift in the predicted dynamics due to a reduced information of the flow dynamics.

This resilience of the POD-DL to a smaller training set is attributed to the dimensionality reduction inherent in the use of the POD, which transforms high-dimensional flow fields into compact, low-dimensional representations, thereby reducing the training complexity compared to models operating directly on full-resolution snapshots.

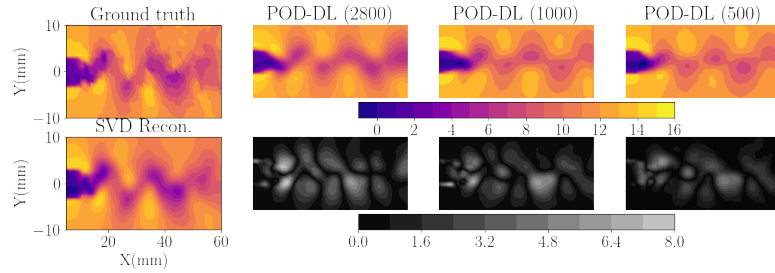
The following section explores a key characteristic of the POD-DL framework, its ability



(a) Ground truth snapshot, predictions and absolute error at time instant 2800.



(b) Ground truth snapshot, predictions and absolute error at time instant 2900.



(c) Ground truth snapshot, predictions and absolute error at time instant 2950.

Figure 3.21: Snapshots at some representative time instants showing the ground truth streamwise velocity component of the turbulent flow, the predictions obtained from the POD-DL, trained with different training data lengths, and their respective absolute error in gray-scale.

to train the deep learning model on POD coefficient vectors. This feature enables the potential reuse of a pre-trained model in entirely different flow scenarios, provided that the number of retained POD modes remains consistent with that of the original training case. Such adaptability presents a compelling advantage in terms of computational efficiency and transferability across different flow conditions.

3.2.4 Results from a pre-trained model

This section presents the results obtained by applying the POD-DL model in a transfer learning context, where the model is pre-trained on a turbulent flow and subsequently tasked with forecasting the dynamics of a distinctly different flow configuration. The approach

involves leveraging previously learned weights without initializing the model from scratch.

Two scenarios are examined: (1) forecasting the new flow dynamics without any additional training on the new dataset, and (2) fine-tuning the pre-trained model using the first 800 snapshots from the new dataset. These cases are designed to evaluate the generalization capability of the POD-DL framework when applied to previously unseen dynamical regimes.

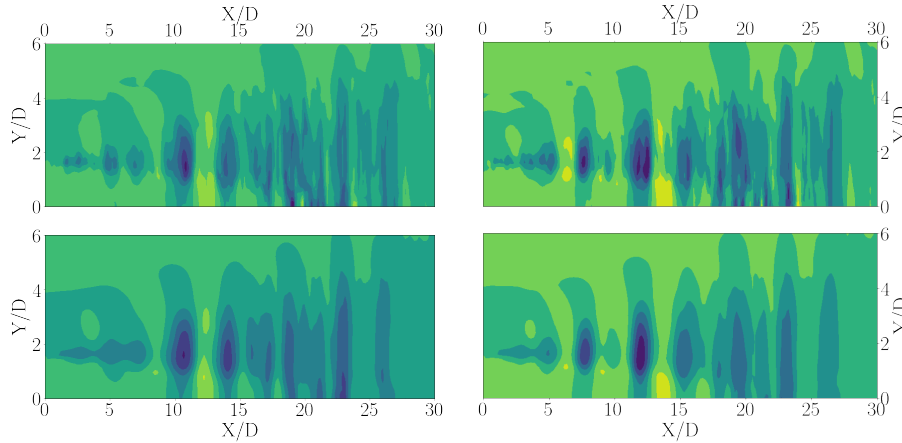


Figure 3.22: Comparison of the original flow from the numerical simulation of the Jet LES (left column) and the reconstruction using SVD (right column), retaining the 20 most energetic modes to reconstruct the flow field, at two different time steps (top and bottom row, respectively).

This task proved infeasible for the RAE and VAE models due to architectural constraints, since both models are rigidly bound to the input dimensions of the training snapshots. Adapting these networks to accommodate snapshots of different dimensions would require substantial reconfiguration of their architectures, thereby rendering the reuse of pre-trained weights impractical. One potential solution to this limitation involves incorporating techniques such as Spatial Pyramid Pooling (He et al., 2014), which enables input resizing while preserving the ability to process spatial hierarchies consistently.

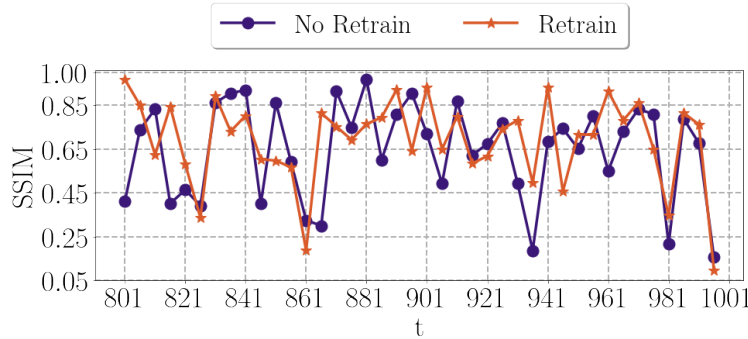


Figure 3.23: Error committed by the POD-DL model with and without retraining, on the stream-wise velocity component of the Jet LES. Errors are quantified using the SSIM defined in (2.23).

In contrast, the POD-DL framework inherently avoids this issue. Since the model operates on POD coefficient vectors rather than full-dimensional snapshots, it is agnostic to the spatial resolution of the original data, provided the number of retained POD modes remains consistent with the training configuration.

The dataset selected to evaluate this generalization capability corresponds to a numerical simulation of an isothermal subsonic jet (Brès et al., 2018) emanating from a round nozzle with an exit diameter of $D = 50$ mm. The simulation was performed using the CharLES large eddy simulation (LES) solver, which solves the spatially-filtered compressible Navier-Stokes equations on unstructured meshes. The solver employs a finite-volume scheme and third-order Runge-Kutta integration for temporal advancement. The simulation utilized approximately sixteen million control volumes and covered a physical time span of 2000 acoustic time units (tc_∞/D) at a Reynolds number of approximately 10^6 .

The corresponding dataset, obtained from this simulation, comprises 5000 snapshots of the streamwise velocity component, sampled every $\Delta t = 0.02$ acoustic time units. The data is extracted on a structured cylindrical grid that closely aligns with the resolution of the LES, extending $30D$ downstream and $6D$ radially to adequately capture the spatial development of the jet.

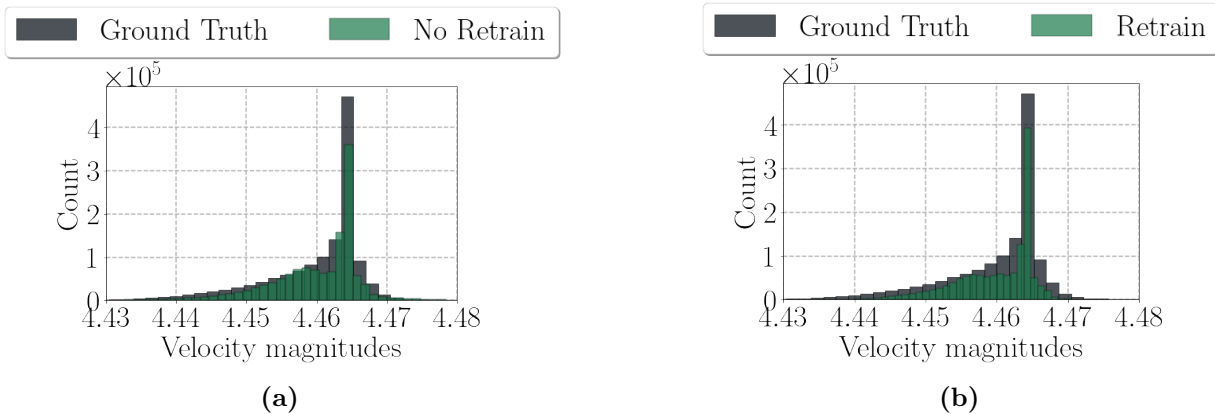


Figure 3.24: Comparison of the velocity histograms between the ground truth and the temporal predictions of the Jet LES obtained from (a) the POD-DL without retraining and (b) the POD-DL with retraining, respectively.

To enable compatibility with the pre-trained POD-DL model, the number of retained POD modes was fixed at 20, matching the original training configuration. Figure 3.22 compares the original jet snapshots with their SVD reconstruction using the 20 most energetic modes.

As shown in Fig. 3.23, the predictive performance of the POD-DL model exhibits negligible differences between the cases with and without retraining. This suggests that retraining with the new dataset does not significantly enhance accuracy, likely due to the dynamical similarities between the POD coefficients obtained from the two flow scenarios. Despite originating from distinct physical configurations, both flows may have POD coefficients with similar behavior, which the neural network appears to generalize effectively. While these findings support the transferability of the POD-DL model across periodic systems, further

investigation is required to assess its performance in other more distinct flow scenarios, which is a direction left open for future research.

Figure 3.24 compares the probability distributions of the ground truth dataset with those predicted by the POD-DL model, providing a statistical measure of predictive fidelity. Interestingly, the model demonstrates slightly improved accuracy when no retraining is performed. This trend is corroborated qualitatively in Fig. 3.25, which presents visual comparisons between ground truth and predicted snapshots.

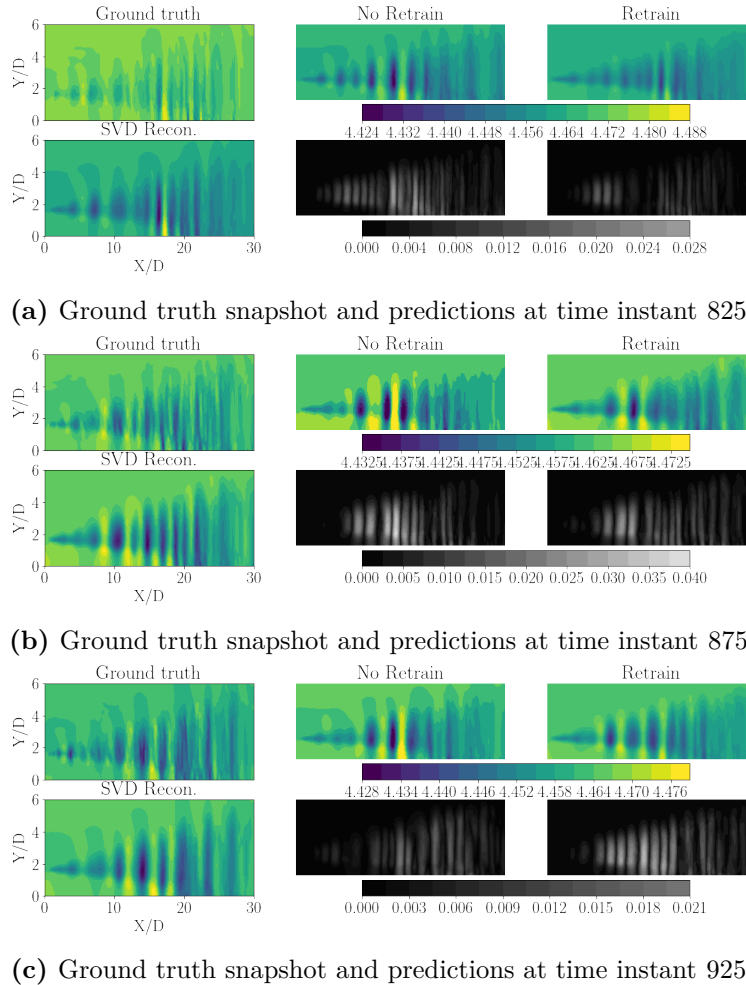


Figure 3.25: Snapshots at some representative time instants showing the ground truth streamwise velocity (left column) and the predictions, from the POD-DL model, for the Jet LES. No retrain (middle column) makes reference to the predictions obtain from case (1) and Retrain (right column) from case (2), described at the beginning of Section 3.2.4.

3.3 Discussion

In this chapter, three forecasting models, POD-DL, residual autoencoder (RAE), and variational autoencoder (VAE), were applied to both laminar and turbulent flow configurations to evaluate their generalization capabilities across varying dynamical complexities. This

evaluation considered the predictive accuracy and the adaptability of each model to diverse datasets. By analyzing their performance across distinct flow types, we assessed each model’s robustness in capturing characteristic fluid behaviors and its practical applicability to a range of scenarios.

From the two test cases involving laminar and turbulent wakes behind a circular cylinder, the models were evaluated in their ability to generate temporally consistent predictions. The results underscored clear differences in generalization and stability. In particular, the VAE consistently underperformed, which may be attributed to the inherent complexity of the ELBO loss function, defined at (3.5), that was designed to approximate a probability distribution. Although VAEs represent a class of probabilistic models, other alternatives, such as temporal diffusion models (Lin et al., 2023) and copula-based methods for multivariate time-series forecasting (Patton, 2013; Smith, 2023), may offer more effective alternatives for capturing such complex dynamics, as shown in Chapter 5. Thus, we do not rule out the potential of probabilistic frameworks for flow prediction, though further research is warranted.

In contrast, both the POD-DL and RAE models utilize the MSE loss function, defined at eq. (2.24), but differ fundamentally in architecture. For instance, the POD-DL model integrates POD with a LSTM network to predict the evolution of low-dimensional POD coefficients, which are then used to reconstruct flow snapshots. Conversely, the RAE is a fully convolutional architecture incorporating ConvLSTM layers to directly predict high-dimensional snapshots without intermediate dimensionality reduction, apart from the convolutional encoder.

A key advantage of the POD-DL framework lies in this two-step approach: it forecasts only the temporal evolution of POD coefficients, significantly reducing input dimensionality. The POD is computed via SVD, which decomposes the dataset into three matrices, $\mathbf{U}$, $\mathbf{\Sigma}$, and $\mathbf{\Gamma}^*$, of which $\mathbf{\Gamma}^*$ is employed during training and forecasting, while $\mathbf{U}$ and $\mathbf{\Sigma}$ are retained for reconstruction. Although this strategy assumes a static $\mathbf{U}$ matrix, which may not fully capture temporal variability, the resulting errors are shown to be minimal. As demonstrated in Figs. 3.7–3.17, the POD-DL model consistently outperforms both RAE and VAE in terms of accuracy and stability for both the laminar and turbulent flows.

The relative underperformance of all models at spatial coordinate P_2 , as shown in Fig. 3.16, highlights the difficulty of modeling small-scale flow structures using architectures like LSTM or ConvLSTM. These networks may lack the capacity to resolve highly localized dynamics. In such contexts, alternative approaches that approximate the statistical behavior of these structures, such as G-LED presented in Chapter 5, could prove more effective.

A fundamental limitation of the RAE model is its architectural dependence on the input resolution. This constraint could potentially be alleviated using techniques like spatial pyramid pooling (He et al., 2014), which enables multi-scale processing of inputs by standardizing their dimensions. However, such solutions introduce additional complexity and reduce architectural flexibility.

In contrast, the POD-DL model benefits from reduced architectural demands by compressing high-dimensional snapshots into low-dimensional vectors that encode the dominant flow dynamics. This dimensionality reduction not only simplifies training but also facilitates the application of traditional multivariate time-series forecasting methods. Furthermore, the

POD-DL framework is highly adaptable across datasets with differing resolutions, as shown in Section 3.2.4, where a pre-trained model effectively forecasted the dynamics of a novel flow without retraining, and with only marginal improvements when fine-tuned.

Nevertheless, the performance of all data-driven models remains contingent on the quality and volume of training data. Predictive accuracy tends to decline with increasing dynamical complexity, especially in turbulent regimes. Among the tested models, the POD-DL approach demonstrated the highest resistance to such conditions. Its effectiveness, however, is sensitive to the number of retained POD modes, while the most energetic modes are well-captured, the dynamics associated with lower-energy and therefore more chaotic modes remain difficult to predict. Capturing these modes may require more expressive architectures than LSTMs, possibly demanding significantly larger training datasets.

In summary, the POD-DL framework emerges as the most robust and versatile among the evaluated models, offering a favorable balance between accuracy, generalization and computational efficiency. Future work should explore the integration of probabilistic or hybrid architectures, such as temporal diffusion models, to improve the forecasting of small-scale and non-periodic flow structures. Moreover, advancing our understanding of the trade-off between training dataset size and model generalization remains a crucial direction for ongoing research.

Chapter 4

An adaptive framework for autoregressive forecasting in CFD using the POD-DL model

As demonstrated in previous chapters, a significant challenge associated with data-driven models lies in managing the divergence of predictions from the reference dynamics, particularly when computed in an autoregressive manner. This divergence often impedes the models' ability to accurately capture the full temporal horizon over which the system is intended to be solved (Abadía-Heredia et al., 2025; Kičić et al., 2023).

The strategies proposed to address this issue can be broadly categorized into two main approaches. The first emphasizes the construction of predictive models that are more resilient to divergence, with diffusion models serving as a prominent example (Lin et al., 2024b). Despite their robustness, such models often demand extensive training datasets (Shysheya et al., 2024) and incur significant computational costs, typically necessitating GPUs for model training.

The second approach prioritizes adaptive methodologies that integrate data-driven models with traditional numerical solvers. These frameworks allow for the deployment of simpler models operating over shorter prediction windows, thereby reducing the need for large training datasets and lowering computational demands. Within this framework, the limitation of a shortened prediction horizon is mitigated by invoking a numerical solver to generate accurate data about the dynamics in regions where the predictive capacity of the model fails. This new data is then used to update the model, enabling it to resume forecasting. (Beltrán et al., 2022; Vega and Le Clainche, 2021).

This chapter concentrates on the second strategy, adaptive frameworks that integrate numerical solvers with DL models to improve forecasting capabilities. Merging DL techniques with numerical solvers has garnered significant attention, especially in the context of high-dimensional and computationally intensive systems. In such scenarios, training a DL model and generating predictions can be substantially more efficient than the time required for a numerical solver to compute even a single time step (Abadía-Heredia et al., 2025; Kičić

et al., 2023). Moreover, traditional solvers often rely on multiple intermediate steps, which are usually not retained, to advance the solution to the desired time point. In contrast, DL models can directly estimate the target time step, bypassing these intermediate stages and thus lowering computational costs.

Several works have explored adaptive approaches of this kind. For example, Kičić et al., 2023 proposed a framework that couples a numerical solver with a DL model to approximate the behavior of diverse dynamical systems. In their approach, the solver initially computes high-fidelity solution data, which is then used to train the DL model. The trained model proceeds to predict future states until a noticeable deviation from the reference dynamics emerges. At that point, the numerical solver is called upon again to compute accurate solutions about the dynamics, which are fed back into the training process. This iterative loop enables the system to adjust dynamically as it evolves over time.

A comparable methodology, combining physics-based simulation tools with machine learning techniques, was presented by Kochkov et al., 2024 in the domain of weather forecasting, where it achieved state-of-the-art accuracy for forecasts extending up to fifteen days.

In an alternative approach Beltrán et al., 2022 substitutes DL models with HODMD (Le Clainche and Vega, 2017), which has been employed to approximate solutions of the complex Ginzburg–Landau equation as well as to forecast intricate phenomena in fluid dynamics (Beltrán et al., 2020).

The principal innovation presented in this thesis, distinguishing it from earlier methods, is the adoption of the hybrid POD-DL model. This model has already been studied in previous chapters and it has shown to improve both predictive accuracy and stability when compared to fully DL alternatives, while also significantly reducing training times, as reported in Chapter 3. Such efficiency is especially valuable in adaptive frameworks aimed at lowering the computational cost associated to traditional numerical solvers.

While the POD-DL model offers enhanced stability relative to purely DL models, its autoregressive nature inherently leads to error accumulation over time, since inaccuracies in early predictions propagate and intensify in later forecasts, as observed in Chapter 3. This limitation underscores the need for an adaptive framework that continuously updates the POD-DL model with newly available data.

4.1 Methods: adaptive framework

As outlined in Chapter 2, the POD-DL model represents spatio-temporal behavior by projecting the dynamics onto a reduced-order basis constructed from POD modes, as expressed in eq. 2.2. This dimensionality reduction targets the most dominant dynamics by retaining only the most energetic modes. Consequently, the system’s complexity is considerably lowered, which in turn supports more stable and extended forecasting.

Nevertheless, a key drawback of the POD-DL model arises from its autoregressive nature: initial prediction errors can propagate and intensify over time, leading to growing inaccuracies in subsequent forecasts. To mitigate this issue, this work explores an adaptive approach

that couples DL forecasting techniques with conventional numerical solvers. In this adaptive framework, updated flow states are periodically computed and used to retrain the DL model, refining its knowledge about the system's behavior. This method not only contributes to improved stability in long-range predictions but also lowers computational demands by reducing the frequency of high-fidelity numerical simulations.

More specifically, the adaptive methodology aims to approximate the solution $\mathbf{v}$ over the domain $\mathcal{P} = (\mathcal{P}x, \mathcal{P}t)$. Initially, the numerical solver is employed to compute high-fidelity solutions over a subset of the temporal domain, where these solutions serve as the initial training data for the DL model, which is described as follows,

1. The numerical solver computes a dataset of solutions, $\mathring{\mathbf{D}} = \mathbf{v}_0, \dots, \mathbf{v}_{T_1}$, corresponding to the time steps $0, \dots, T_1$, where $T_1 \ll T$.
2. The dataset $\mathring{\mathbf{D}}$ serves as the training data for a DL forecasting model, $\mathcal{M}$, which is then used to predict the system's evolution over the following time steps, i.e., $\mathcal{M}(\mathring{\mathbf{D}}) = \mathbf{v}_{T_1+1}, \dots, \mathbf{v}_{T_1+T_2}$.
3. The numerical solver is subsequently called to compute an additional set of solutions, $\mathring{\mathbf{D}} = \mathbf{v}_{T_1+T_2+1}, \dots, \mathbf{v}_{T_1+T_2+T_3}$.
4. Steps (2) and (3) are repeated iteratively until the solution spans the full temporal domain up to time T .

As this chapter represents an initial investigation into the adaptive framework, the primary focus is on evaluating its potential to address complex problems in fluid dynamics. Within this scope, this chapter aims to assess both the theoretical computational speed-up the framework may provide and the challenges faced by the underlying DL forecasting model.

Therefore, instead of employing a live numerical solver, it is substituted with a precomputed dataset. Each time the methodology calls for new solutions from the numerical solver (step 3), they are retrieved from the precomputed dataset. This modification allows to focus on evaluating the performance of the DL forecasting model in this adaptive framework without the additional computational overhead of real-time numerical computations. The integration of a live numerical solver is left open for future research.

Therefore, rather than utilizing a live numerical solver, a precomputed dataset is used as a surrogate. Each time the methodology requires new solutions from the solver (step 3), these are drawn from this existing dataset. This adjustment enables the evaluation of the DL forecasting model's performance within the adaptive framework, while avoiding the computational cost of running a numerical simulation. The integration of a live numerical solver is deferred to future work.

The following section offers a comprehensive overview of the DL forecasting model employed in this study, detailing its underlying motivation, architectural design and the training methodology adopted.

4.2 Forecasting deep learning model

The forecasting DL model, integrated within the POD-DL framework, presented in Chapter 2, and whose architecture is summarized in Tab. 4.1, is designed to capture temporal dependencies over a window of length m . This window is constructed from a sequence of past POD coefficients, $\gamma^{s-m+1}, \dots, \gamma^s$, and used to predict the subsequent state, γ^{s+1} . In this context, $\gamma^s \equiv \mathbf{\Gamma}_{s,1:k}$ denotes the s -th snapshot of the first k retained coefficients.

As noted in eq. (2.2), the number of retained modes satisfies $k \leq K$, where K is the total number of POD modes. Substituting k into eq. (2.2) yields an approximate representation of the original field, since the contribution of modes beyond the first k is omitted. This truncation results in a loss of information, as part of the original system dynamics is no longer captured.

To perform the prediction, the model employs a LSTM architecture (Hochreiter and Schmidhuber, 1997) as its initial layer. This architecture is characterized by the presence of three gating mechanisms: the input gate, the forget gate and the output gate. These gates govern the flow of information through the network by determining how input data are processed over time. Specifically, the input gate determines which new information to incorporate, the forget gate discards information deemed no longer relevant and the output gate regulates the information passed to the next cell in the sequence.

In this chapter, the input, forget and output gates of the LSTM architecture are denoted by $\mathbf{i}$, $\mathbf{f}$ and $\mathbf{o}$, respectively. The cell state is represented by $\mathbf{c}$ and the hidden state by $\mathbf{h}$. The computation of these components follows the standard LSTM formulation, as detailed in eqs. 2.13-(2.18).

To increase the model's flexibility, the final hidden state produced by the LSTM, which encodes the time-ahead prediction, is passed through a sequence of three fully connected (dense) layers. This post-processing step yields the final forecast of the POD coefficients.

Table 4.1: Architectural specifications of the POD-DL model. Here, k denotes the number of retained POD modes, m is the input window size (i.e., the number of past time steps used for forecasting) and B indicates the batch size used during training.

Layer	Name	# Neurons	Activation function	Dimension
0	Input	-	-	$B \times m \times k$
1	LSTM	100	ReLU	$B \times 1 \times 100$
2	Dense	200	ReLU	$B \times 1 \times 200$
3	Dense	200	Linear	$B \times 1 \times 200$
4	Dense	k	Linear	$B \times 1 \times k$

Prior to training, the dataset comprising the POD coefficients is standardized using the mean and standard deviation computed from the entire available dataset. Although the model functions autoregressively during inference, it is trained using a one-step-ahead prediction strategy. In this approach, the model predicts the next state in the sequence and directly compares it to the corresponding ground truth. This training procedure is implemented via a rolling window technique applied as a preprocessing step.

As detailed in Appendix A, the rolling window method reformulates the sequence of POD coefficients into structured input-output pairs suitable for supervised learning. For a dataset of temporally ordered coefficients, each training sample is constructed by selecting a fixed-length window of m consecutive coefficients as input, followed by the next coefficient in the sequence as the target output. The window is then shifted forward by one time step to create the subsequent input-target pair. Repeating this process across the dataset allows the model to learn temporal correlations effectively by mapping past states to future predictions. This is visually depicted at Fig. A.1 in the Appendix A.

The window size m used in the training process is influenced by both the characteristics of the flow dynamics and the architecture of the DL model, as discussed in Abadía-Heredia, Corrochano, et al., 2024. In accordance with this prior work, a fixed window size of $m = 10$ is employed for all test cases presented in this chapter.

Although the windows are generated sequentially from the time-series data, they are randomly shuffled before training to ensure statistical independence among training samples. Consistent with standard practices in machine learning, these windows are further organized into mini-batches to enhance computational efficiency during training. The batch size B is usually set to a power of two, with $B = 8$ used uniformly across all test cases discussed in Section 4.4.

To prevent issues such as exploding gradients or vanishing activations, all model weights are initialized using strategies tailored to the specific activation functions and layers: Xavier (Glorot and Bengio, 2010), Kaiming (He et al., 2015) or orthogonal initialization. This activation-dependent initialization approach is designed to maintain the variance of signals throughout the network layers, as recommended in Goodfellow et al., 2016. Details on the initialization scheme can be found in the publicly available code from Abadía-Heredia et al., 2025.

Model training is performed using the Adam optimizer (D. P. Kingma and Ba, 2017), with the recommended parameter values: $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$, as specified in the original publication.

The model is trained by minimizing the MSE loss function, defined as follows,

$$\text{MSE} = \frac{1}{S} \sum_{i=0}^{S-1} (\gamma^i - \hat{\gamma}^i)^2, \quad (4.1)$$

where $\hat{\gamma}^i$ is the i -th prediction obtained from the POD-DL model.

To reduce the risk of the loss function becoming trapped in local minima, this work employs a learning rate scheduler known as CosineAnnealingWarmRestarts (Loshchilov and Hutter, 2017). This strategy gradually decreases the learning rate from a specified maximum to a minimum over a predefined number of epochs, shorter than the total training duration. After each decay phase, the learning rate is reset to its maximum value, initiating a new decay cycle. In the present implementation, each subsequent cycle spans twice the number of epochs as the previous one. This periodic reset mechanism encourages better convergence by enabling the optimizer to escape suboptimal regions of the loss landscape. An example of the evolution

of the learning rate over time, along with the corresponding loss function values, is shown in Fig. 4.1.

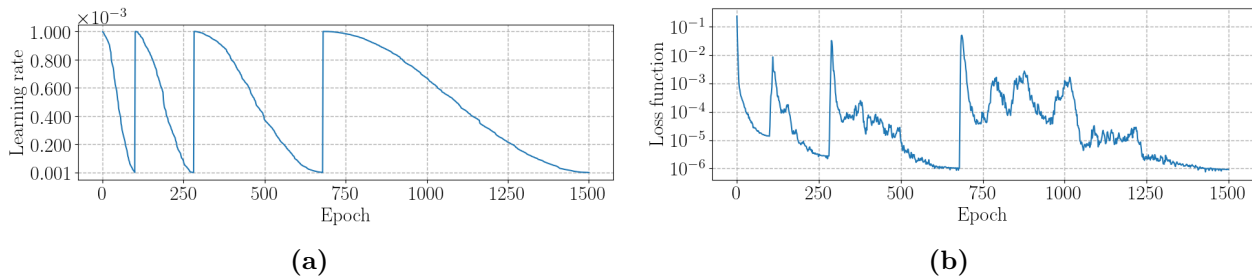


Figure 4.1: (a) Evolution of the learning rate over 1500 epochs under the CosineAnnealingWarmRestarts schedule, oscillating between 10^{-3} and a minimum value of 10^{-6} . (b) Corresponding training loss, showing pronounced decreases immediately after each warm restart as the learning rate returns to its peak value.

The following section offers a detailed explanation of the implementation of the adaptive framework introduced in Sec. 4.1, highlighting the key hyperparameters that govern its performance and the metric used to evaluate predictive accuracy.

4.3 Implementation of the adaptive framework

The adaptive framework outlined in Sec. 4.1 depends on several hyperparameters that influence both the quality of the results and the efficiency of training the DL model. Among these, a subset has been identified as particularly influential to the framework’s overall performance and is summarized in Tab. 4.2.

Table 4.2: Key hyperparameters governing the behavior of the adaptive framework. A brief description of each hyperparameter is provided in the left column, with the corresponding symbol used throughout the text listed in the right column.

Hyperparameter	Symbol
Training length (Epochs)	E
Number snapshots for the first training	S_0
Number of snapshots for subsequent trainings	S_1
Number of predictions	P

These hyperparameters were identified through extensive experimentation. The parameters found to have the greatest impact on the performance of the adaptive framework include the number of snapshots used for initial training, S_0 , those used in subsequent retraining phases, S_1 , and the number of predictions between retrainings, P . In contrast, variations in the training duration, E , were observed to have minimal influence on the model’s overall accuracy. A schematic representation of the adaptive framework developed in this work is presented in Fig. 4.2.

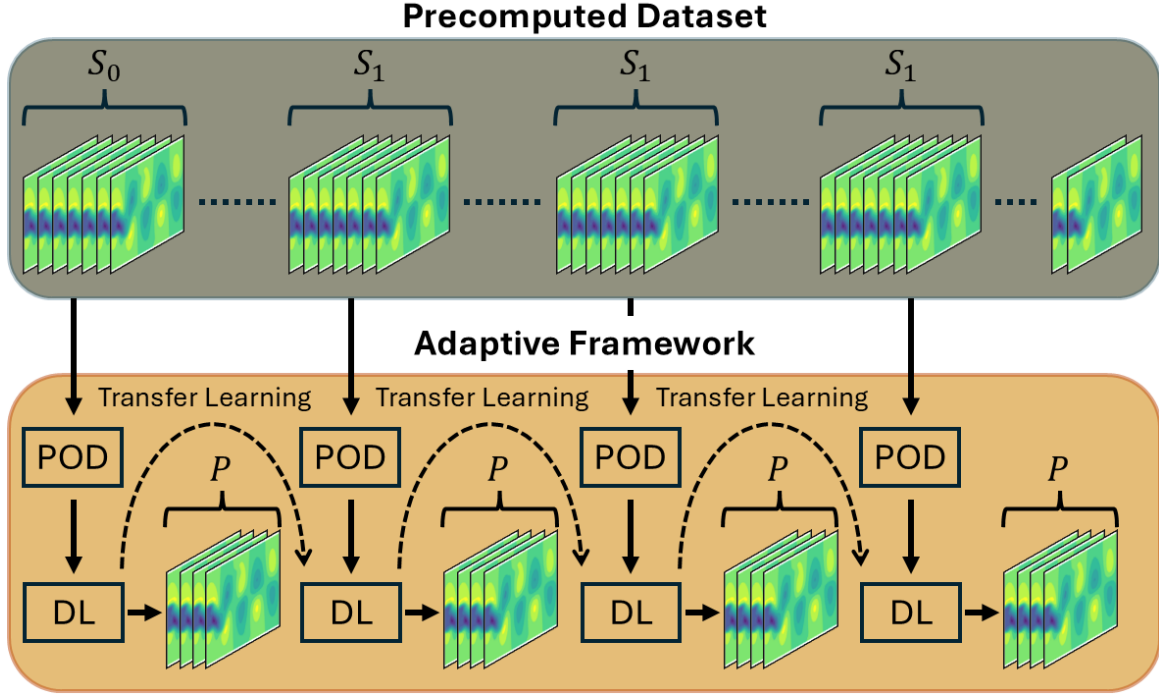


Figure 4.2: Schematic representation of the adaptive framework proposed in this chapter. Initially, S_0 snapshots are extracted from a precomputed dataset to train the POD-DL model. POD is applied to obtain the corresponding coefficients, which are then used to train a DL model over E epochs. The trained model forecasts P future snapshots, after which an additional set of S_1 snapshots is retrieved for retraining. During each retraining cycle, the DL model employs transfer learning by reusing the weights from the previous training phase. This process is iteratively repeated until predictions span the entire temporal domain of interest.

As described in Section 4.1, the adaptive framework begins with an initial set of snapshots, extracted from the precomputed dataset, to perform the first training of the POD-DL model introduced in Sec. 4.2. The size of this initial training set, denoted by S_0 , varies depending on the specific problem and, more critically, on the complexity of the underlying flow dynamics. As will be shown, laminar flows typically require fewer snapshots for effective training compared to turbulent flows. This distinction reflects the greater complexity and variability inherent in turbulent regimes, which demand a larger dataset to adequately capture their behavior.

If the combined number of training snapshots and predicted snapshots does not fully span the temporal domain of interest, an additional subset of data, denoted by S_1 , is drawn from the precomputed dataset to retrain the POD-DL model. This quantity, S_1 , remains constant across all subsequent retraining steps.

Each retraining phase is conducted exclusively on the new set of S_1 snapshots, with earlier data discarded. To mitigate the limitations of training on a reduced dataset, the adaptive framework incorporates a transfer learning strategy: each retraining session initializes the model with the weights obtained from the preceding training phase. This approach ensures continuity in the learning process and enhances training efficiency. The effectiveness of

transfer learning in the POD-DL model has previously been demonstrated in Abadía-Heredia, Corrochano, et al., 2024.

The number of training epochs, E , is influenced by both the dynamical complexity of the system and the learning rate schedule discussed in Section 4.2. In particular, it is advantageous for training to conclude at the end of a learning rate decay cycle, thereby allowing the learning rate to reach its minimum, as illustrated in Figure 4.1, and promoting optimal convergence.

In this chapter, the trained POD-DL model consistently generates a fixed number of predictions, P , after each training or retraining phase. The integration of uncertainty quantification to assess the reliability of these predictions is deferred to future work. Nevertheless, several approaches may be considered, including bootstrap techniques (Kičić et al., 2023; Lakshminarayanan et al., 2017), POD-based structural methods (Abadía-Heredia, Lopez-Martín, and le Clainche, 2024; Beltrán et al., 2022; Rapún et al., 2015), conformal prediction frameworks (Auer et al., 2023) or evaluation of the residuals of the governing partial differential equations (Rapún et al., 2015).

To assess prediction reliability in this chapter, we adopt a metric referred to as the probability of prediction error (PPE), where it represents the probability that, for a randomly selected spatial coordinate in a predicted snapshot, the absolute error falls below a specified threshold $\alpha > 0$. It is formally defined as follows,

$$\Pr_{\alpha}(t_S) \equiv \Pr(-\alpha \leq \mathbf{v}_s - \hat{\mathbf{v}}_s \leq \alpha). \quad (4.2)$$

Here, $\mathbf{v}_s$ and $\hat{\mathbf{v}}_s$ denote the s -th ground truth and predicted snapshots, respectively. The selection of the error threshold α must be informed based on the specific characteristics and scales inherent to the problem under consideration. The pseudocode used to compute this probability is provided at the Algorithm 1 in Appendix B.

A high value of PPE for a relatively small error threshold, when evaluated in the context of the system’s dynamics, indicates higher predictive accuracy. Thus, it is desirable for this probability to approach unity, i.e., $\Pr_{\alpha}(t_S) \simeq 1$.

Given that the primary goal of this chapter is to develop an adaptive framework that reduces the computational cost of solving dynamical systems relative to conventional numerical solvers, the following metric is introduced to estimate the theoretical computational saving (TCS) achieved,

$$\text{TCS} = (1 - (S_0 + nS_1)/S) \times 100\%. \quad (4.3)$$

Here, n denotes the number of times additional data were extracted from the precomputed dataset and S represents the total number of available snapshots, as defined in eq. (5.2). The TCS metric provides a percentage of the theoretical reduction in computational cost, quantifying the savings achieved by employing the adaptive framework instead of computing all snapshots using a traditional numerical solver.

This adaptive framework was evaluated on three distinct datasets, each characterizing the

dynamics of different flow conditions. The following section presents the results for each case, obtained under varying values of S_0 , S_1 and P .

4.4 Results

As described in Section 4.1, the adaptive framework examined in this study assumes the availability of a precomputed dataset from which high-fidelity data are drawn to train the POD-DL model. Specifically, three datasets, introduced in previous chapters, are utilized: the three-dimensional wake behind a circular cylinder under both laminar and turbulent regimes and an isothermal subsonic jet in a turbulent regime.

For the remainder of this work, and to maintain clarity, the datasets corresponding to the circular cylinder wake in laminar and turbulent regimes will be referred to as the laminar flow and turbulent flow, respectively.

4.4.1 Three-dimensional wake behind a circular cylinder at laminar regime: laminar flow

The dataset corresponding to the laminar flow past a cylinder was already introduced in Chapter 2, it was originated from the DNS performed by Le Clainche et al., 2018, who solved the incompressible Navier–Stokes equations, defined in eqs. (2.20)-(2.28), at a Reynolds number $Re = 280$ with the spectral element-Fourier solver *Semtex* (Blackburn et al., 2019). The computational domain extends from $15D$ to $50D$ downstream of the cylinder center, $D = 1$, $-15D \leq y \leq 15D$ in the wall-normal direction and a (non-dimensional) spanwise period of length $L_z \approx 6.99D$. In the (x, y) planes the flow is represented by 600 ninth-order macro-elements, while the homogeneous z -direction is discretized by 64 equispaced Fourier modes. The boundary condition are uniform in the inflow, the outflow pressure is set to zero and periodic conditions are set close to the spanwise boundaries.

After the initial two-dimensional stage the von Karman vortex street reaches its limit cycle, where three-dimensional disturbances grow naturally and saturate the flow. During this development the simulation is sampled every non-dimensional time unit, yielding a sequence of 499 uniformly spaced snapshots that spans roughly 500 convective time units. Each snapshot is interpolated onto a structured grid of $100 \times 40 \times 64$ spatial points and stores the streamwise and wall-normal velocity components only, giving the following tensor shape for the dataset,

$$\dot{\mathbf{D}} \in \mathbb{R}^{2 \times 100 \times 40 \times 64 \times 499}. \quad (4.4)$$

This dataset captures the fully laminar three-dimensional regime, and serves as the baseline for the evaluation of the adaptive framework.

Due to the relatively simple dynamics of this flow, the POD-DL model’s performance in this case shows limited sensitivity to the selection of hyperparameters. Three experiments were performed on this dataset, varying the values of S_1 and P , while keeping the initial training

size fixed at $S_0 = 100$, a very small fraction of the total snapshot set. The complete list of hyperparameters used in these experiments is presented at Tab. 4.3.

Table 4.3: Hyperparameter settings used to evaluate the adaptive framework on the laminar flow around a circular cylinder. In the second test case, the number of predictions P is increased from $P = 100$ to $P = 200$ after the second training phase.

Test	P	S_0	S_1	E
T1	100	100	50	1500
T2	100/200	100	100	1500
T3	400	100	50	1500

The adaptive framework is applied to the velocity field, which is decomposed into its corresponding POD modes and coefficients using the procedure outlined in eq. (2.10). The DL model is subsequently trained on the resulting POD coefficients. During the initial training phase, based on S_0 snapshots, it is essential to determine the number of POD modes to retain for effective forecasting. This choice is typically guided by the proportion of energy captured by the retained modes, quantified through the cumulative energy defined in eq. (2.25).

Figure 4.3 shows the cumulative energy as a function of the truncation level, computed by applying POD to the first S_0 snapshots of the laminar flow. For flows of this type, it is common practice to retain approximately 90% of the total energy in the dynamics. In this case, achieving this threshold corresponds to retaining the first 6 POD modes.

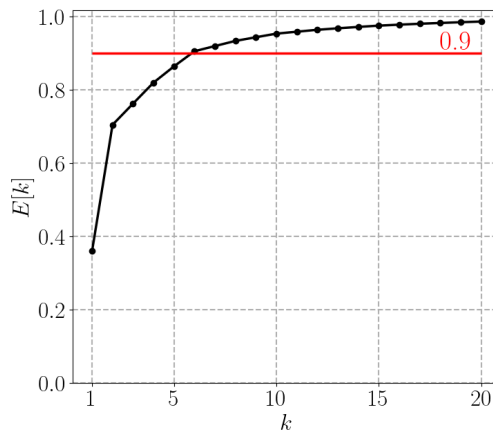


Figure 4.3: Percentage of cumulative energy, $E(k)$, retained when truncating at the k most energetic modes (up to $k = 20$) for the laminar flow.

This truncation level is maintained for all subsequent training phases within the adaptive framework. Consequently, it is necessary to ensure that $S_1 \geq k$, otherwise, the number of POD modes computed from the updated set of S_1 snapshots would be insufficient to preserve the selected truncation level k .

Prediction reliability will be evaluated using the probability of prediction error (PPE) metric, as defined in eq. (4.2), complemented by percentile-based comparisons. Notably, this assessment

is not performed directly on the velocity field but rather on the vorticity field, which is derived from the velocity components according to eq. (2.32).

This choice is motivated by the fact that the vorticity field offers a more intuitive representation of the underlying flow physics. In particular, it enables a clearer assessment of how well the POD-DL model captures the location and structure of vortical features by facilitating direct comparison with the reference solution obtained from high-fidelity numerical simulations.

Accordingly, Fig. 4.4 provides a visual comparison between predicted and reference vorticity fields at selected time steps. The observed vorticity magnitudes predominantly lie within the absolute interval $[7, 16]$. Within this range, an absolute prediction error of 1 corresponds to a relative error of approximately 6.25% to 14.2%, while an error of 0.5 falls within 3.12% to 7.14%. To ensure that the POD-DL model remains consistent with the reference dynamics, it is desirable to attain high probabilities of absolute prediction errors falling within these thresholds.

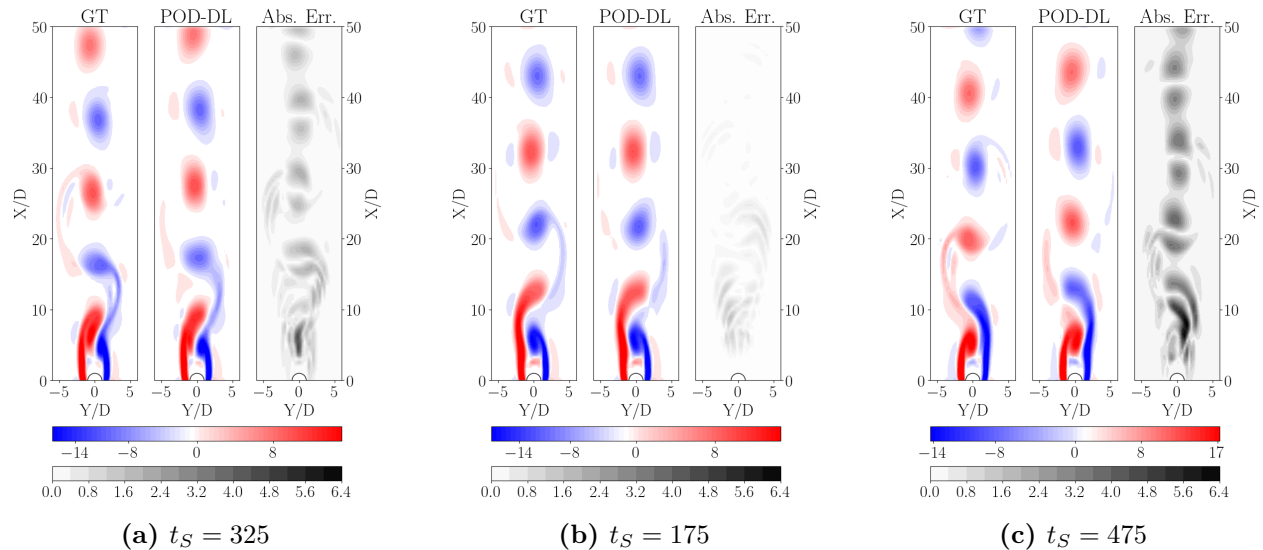


Figure 4.4: Comparison of ground truth (GT) and POD-DL predicted vorticity fields for the laminar flow at representative time steps from test cases T1 (a), T2 (b) and T3 (c). The index t_S refers to the snapshot index, not the physical time t .

Figure 4.5 presents the probability of achieving absolute vorticity prediction errors less than or equal to 0.5 and 1, respectively, for the three test cases listed in Tab. 4.3. These probabilities are quantified using the PPE metric and are given by $\text{Pr}\alpha(t_S) \equiv \text{Pr}(-\alpha \leq \omega_s - \hat{\omega}_s \leq \alpha)$ for $\alpha \in \{0.5, 1\}$, where ω_s and $\hat{\omega}_s$ denote the s -th ground truth and predicted vorticity snapshots, respectively. Notably, snapshots where the probability reaches unity ($\text{Pr}\alpha(t_S) = 1$) correspond to those computed by the numerical solver, and included in the model’s training.

The figure also displays the evolution of the spatially averaged streamwise velocity, serving as a visual reference for the dynamic behavior captured by the snapshots used in both the initial and subsequent training phases of the POD-DL model.

As illustrated in Figs. 4.5(a) and (c), corresponding to test cases T1 and T3, a notable effect associated with a distributional shift in the flow dynamics can be observed around snapshot

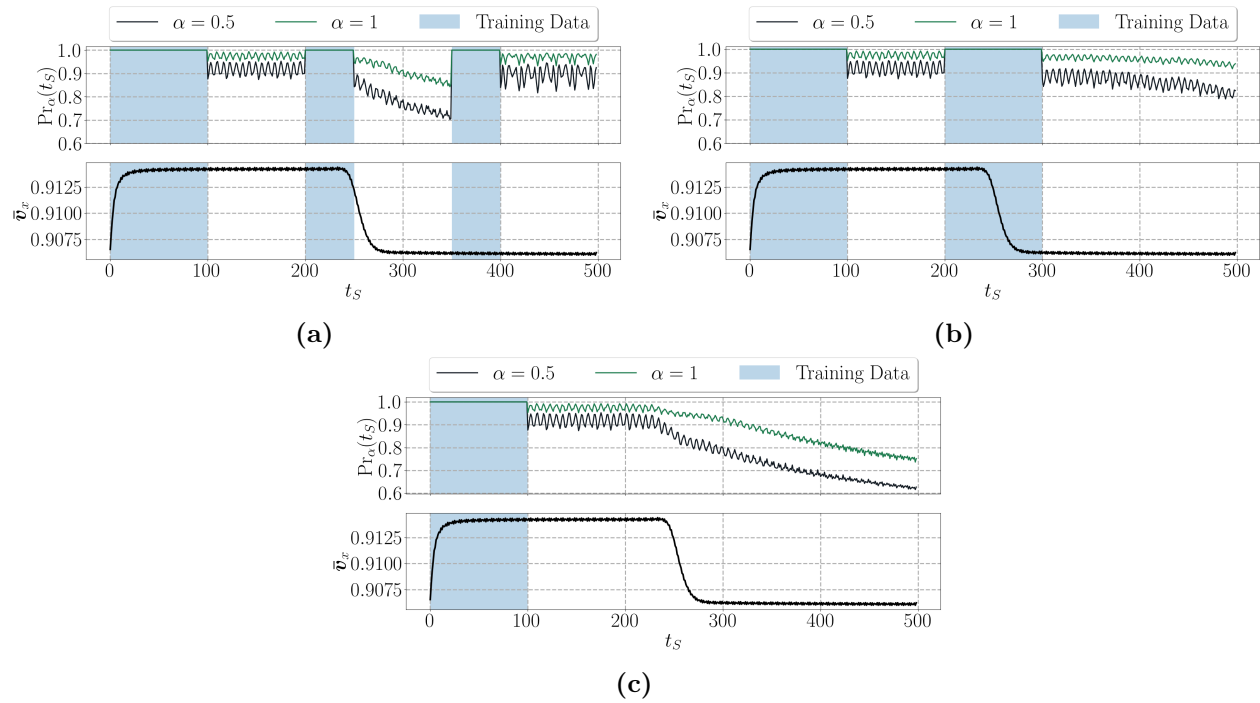


Figure 4.5: The top row shows the probability of achieving an absolute vorticity prediction error below thresholds of 0.5 (orange) and 1 (green) for the three test cases: T1 (a), T2 (b) and T3 (c), in the laminar flow. The bottom row illustrates the temporal evolution of the spatially averaged streamwise velocity. The selected error thresholds are based on the typical vorticity range observed in the flow, as indicated in Fig. 4.4.

250. In both cases, the probability of achieving low vorticity prediction errors begins to decrease precisely at the onset of this shift, as highlighted in the bottom panels of each figure. This trend indicates that the model’s predictions start to deviate from the reference solution as the underlying dynamics evolve.

In contrast, Fig. 4.5 (b) demonstrates how this issue is mitigated in test case T2. Here, the incorporation of new training data that reflect the distributional change enables the POD-DL model to adapt during retraining, resulting in predictions that remain consistent with the altered dynamics.

Even in test case T1, where divergence from the reference dynamics is initially observed at the point of distributional change, retraining with a relatively small number of new snapshots effectively restores predictive accuracy. This underscores the utility of transfer learning within the adaptive framework, as the model is able to leverage knowledge from previous training phases to accommodate new dynamical regimes with minimal additional data.

Further evidence of this behavior is provided in Fig. 4.6, which compares the 0.25-th, 0.5-th (median) and 0.75-th percentiles of the ground truth vorticity field with those predicted by the adaptive framework across all time steps. Consistent with earlier observations, predictions in test cases T1 and T3 begin to diverge near snapshot 250, corresponding to the distributional shift. In contrast, this divergence is not observed in test case T2, where retraining successfully

incorporates the updated dynamics. Additionally, in T1, the initial deviation is corrected once the retrained model integrates representative samples of the new flow behavior, resulting in renewed alignment with the reference solution.

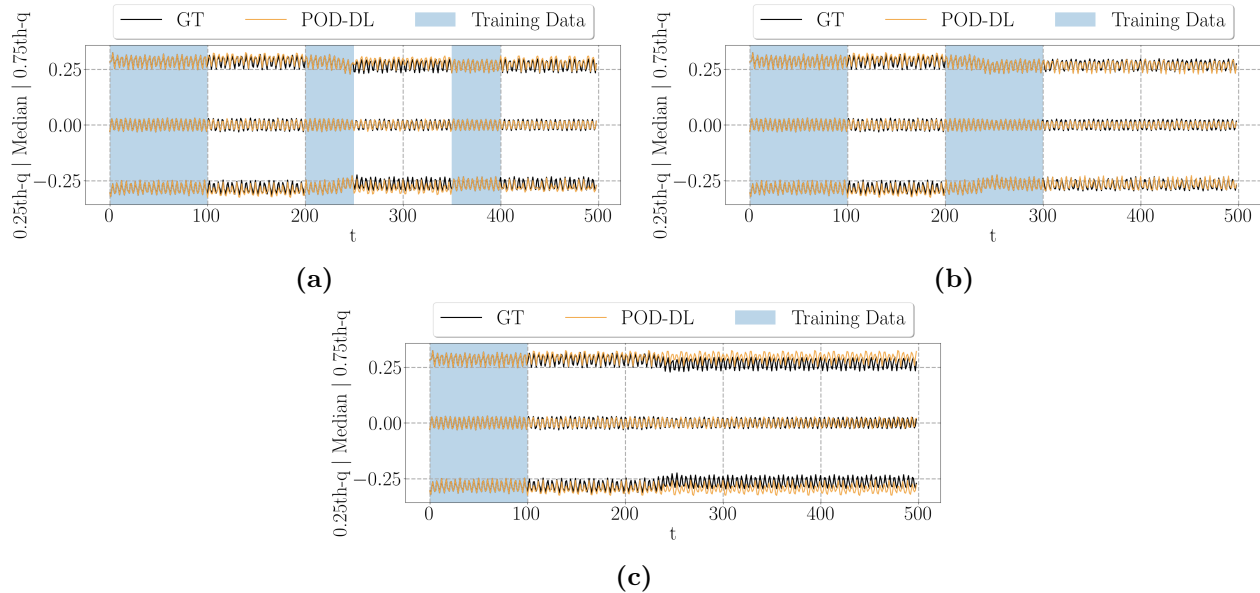


Figure 4.6: Temporal evolution of the 0.25-th (bottom), 0.5-th (middle) and 0.75-th (top) vorticity percentiles for the ground truth (blue) and POD-DL predictions (orange) across test cases T1 (a), T2 (b) and T3 (c) in the laminar flow scenario. Shaded regions denote the time intervals corresponding to snapshots used during both initial training and subsequent retraining phases.

Following extensive experimentation, it was found that training the POD-DL model using 6 retained POD modes and a training set of 100 snapshots requires approximately 1.2 minutes on an Intel(R) Core(TM) i7-10700 CPU @ 2.90GHz with 16 cores and 64 GB of RAM. The computational cost associated with making predictions is virtually negligible, averaging around 2 seconds regardless of the number of predicted time steps.

Under these conditions, the theoretical computational savings, quantified using the TCS metric defined in eq. (4.3), are estimated to be approximately 60% for test cases T1 and T2, where 299 out of 499 time steps were predicted. For test case T3, in which 399 time steps were predicted, the theoretical saving reaches 80%, although this case exhibits the lowest prediction accuracy among the three. Nonetheless, these results collectively highlight the potential of the adaptive framework to achieve significant computational efficiency in modeling laminar flows.

4.4.2 Three-dimensional wake behind a circular cylinder at turbulent regime: turbulent flow

Similar to the laminar flow case discussed in Section 4.4.1, the dataset examined in this section also represents the wake of a circular cylinder. The key distinction, however, is that it corresponds to a turbulent regime, characterized by a Reynolds number of $Re = 4000$.

The turbulent cylinder dataset $\tilde{\mathbf{D}}$ is a subset of the time-resolved PIV experiment from Mendez et al., 2019. In the experimental campaign, a smooth cylinder of diameter $D = 5$ mm was tested in the L10 low-speed wind tunnel at the von Kármán Institute. With a free-stream velocity $U_\infty = 12.1$ m/s the flow reaches a $Re \approx 4 \times 10^3$, within the fully turbulent, three-dimensional shedding regime. The wake was imaged in a 70×26 mm plane centred 2D downstream of the cylinder using a Photonics Industries DM20-527 Nd:YLF laser (3 kHz sampling rate) and an Optronis SpeedSense 9090 CMOS camera. After adaptive processing in the DYNAMICSTUDIO software the final window was 24×24 px with 50% overlap, giving a vector grid spacing of $\Delta x = \Delta y \approx 0.66$ mm and a sampling rate of $f_s = 3$ kHz.

The dataset used in this section exclusively captures the initial steady-state flow at $Re = 4000$. The subsequent transition through a transient regime and eventual stabilization at $Re = 2600$, as detailed in Section 3.2.2, are not included.

To evaluate the framework’s performance under highly complex dynamics, several configurations of the adaptive framework were tested by varying key hyperparameters. These configurations are summarized in Tab. 4.4. As in the previous case, the number of initial training snapshots was small relative to the full dataset, with $S_0 = 200$.

Table 4.4: Hyperparameters used to test the adaptive framework on the turbulent flow.

Test	P	S_0	S_1	E
$T1$	50	200	100	1500
$T2$	100	200	100	1500
$T3$	100	200	200	1500
$T4$	500	200	200	1500

As with the laminar flow case, POD is applied to the first $S_0 = 200$ snapshots to compute the corresponding POD coefficients. The truncation level for the POD modes is determined based on the fraction of total energy retained by these modes, as defined by the cumulative energy function in eq. (2.25). Previous studies (A. Martínez-Sánchez et al., 2023) have indicated that, in similar turbulent flows, capturing approximately 30% of the total energy is often sufficient to preserve the principal flow dynamics of interest. As shown in Fig. 4.7, this energy threshold is met by retaining the first 13 POD modes.

This truncation level is preserved across all training iterations within the adaptive framework to ensure consistency with the POD-DL model architecture, which remains unchanged throughout the process. As in the laminar case, the POD-DL model is trained using velocity field snapshots, while the evaluation of prediction accuracy is conducted on the derived vorticity field, computed as defined in eq. (2.32).

For the turbulent flow scenario, predictive performance is assessed using the probability of prediction error (PPE) metric, as introduced in eq. (4.2). This evaluation is further complemented by percentile-based comparisons and an analysis of the turbulent kinetic energy (TKE). The TKE is defined as half the sum of the variances of the fluctuating components of the velocity field and is mathematically expressed as,

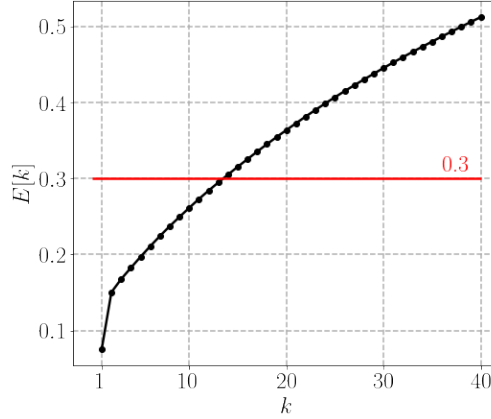


Figure 4.7: Cumulative energy percentage, $E(k)$, retained when truncating at the k most energetic modes (up to $k = 40$) for the turbulent flow.

$$\text{TKE} = \frac{1}{2} \left(\overline{(\tilde{v}_x)^2} + \overline{(\tilde{v}_y)^2} \right), \quad (4.5)$$

$$\overline{(\tilde{v}_x)^2} = \frac{1}{T} \int_0^T \tilde{v}_x(t)^2 dt = \frac{1}{T} \int_0^T (\mathbf{v}_x(t) - \bar{\mathbf{v}}_x)^2 dt, \quad (4.6)$$

$$\bar{\mathbf{v}}_x = \frac{1}{T} \int_0^T \mathbf{v}_x(t) dt. \quad (4.7)$$

Here, $\tilde{\mathbf{v}}_x$ denotes the streamwise velocity fluctuations, computed by subtracting the time-averaged mean flow from the instantaneous velocity, i.e., $\tilde{\mathbf{v}}_x(t) = \mathbf{v}_x(t) - \bar{\mathbf{v}}_x$. The same procedure is applied to the wall-normal velocity component, $\mathbf{v}_y$, to isolate its fluctuating part. TKE serves as a diagnostic measure to assess how well the predicted dynamics produced by the adaptive framework align with the underlying physical behavior of the turbulent flow.

As in the laminar case, Fig. 4.8 presents a visual comparison between predicted and reference vorticity fields at selected time steps. For the turbulent wake, vorticity values primarily lie within the absolute range $[2.5, 7]$, which is narrower than the range observed in the laminar flow. Based on this interval, an absolute prediction error of 0.5 corresponds to a relative error range of $[7.1\%, 20\%]$, while an error of 0.2 corresponds to $[2.8\%, 8\%]$. These error magnitudes are considered moderate for a turbulent regime, especially considering that the POD-DL model focuses on coherent structures by retaining only the most energetic modes.

Figure 4.9 displays the probability of achieving absolute vorticity prediction errors less than or equal to 0.2 and 0.5, respectively, for the four test cases listed in Tab. 4.4. This probability is quantified using the PPE metric, as defined in eq. (4.2), and is expressed as $\text{Pr}\alpha(t_s) \equiv \text{Pr}(-\alpha \leq \boldsymbol{\omega}_s - \hat{\boldsymbol{\omega}}_s \leq \alpha)$ for $\alpha \in \{0.2, 0.5\}$, where $\boldsymbol{\omega}_s$ and $\hat{\boldsymbol{\omega}}_s$ denote the s -th ground truth and predicted vorticity snapshots, respectively.

Note that, in the turbulent flow case, the spatially averaged streamwise velocity ($\bar{\mathbf{v}}_x$, shown in the bottom row) exhibits sharper and more abrupt temporal variations. This behavior is characteristic of the high complexity and strongly fluctuating nature of turbulent dynamics.

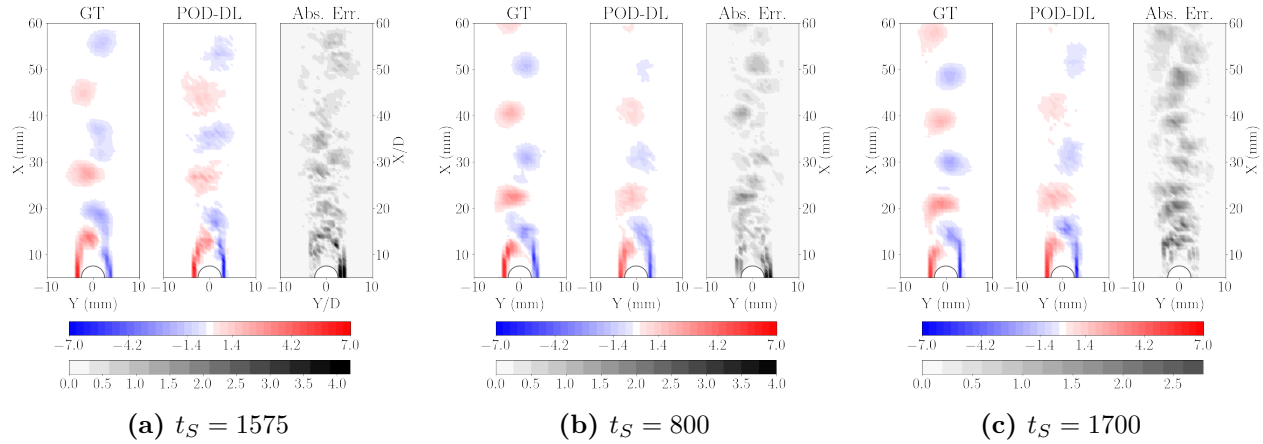


Figure 4.8: Comparison of ground truth (GT) and POD-DL predicted vorticity fields for the turbulent flow at representative time steps from test cases T1 (a), T3 (b) and T4 (c). The index t_S refers to the snapshot number, not the physical time t .

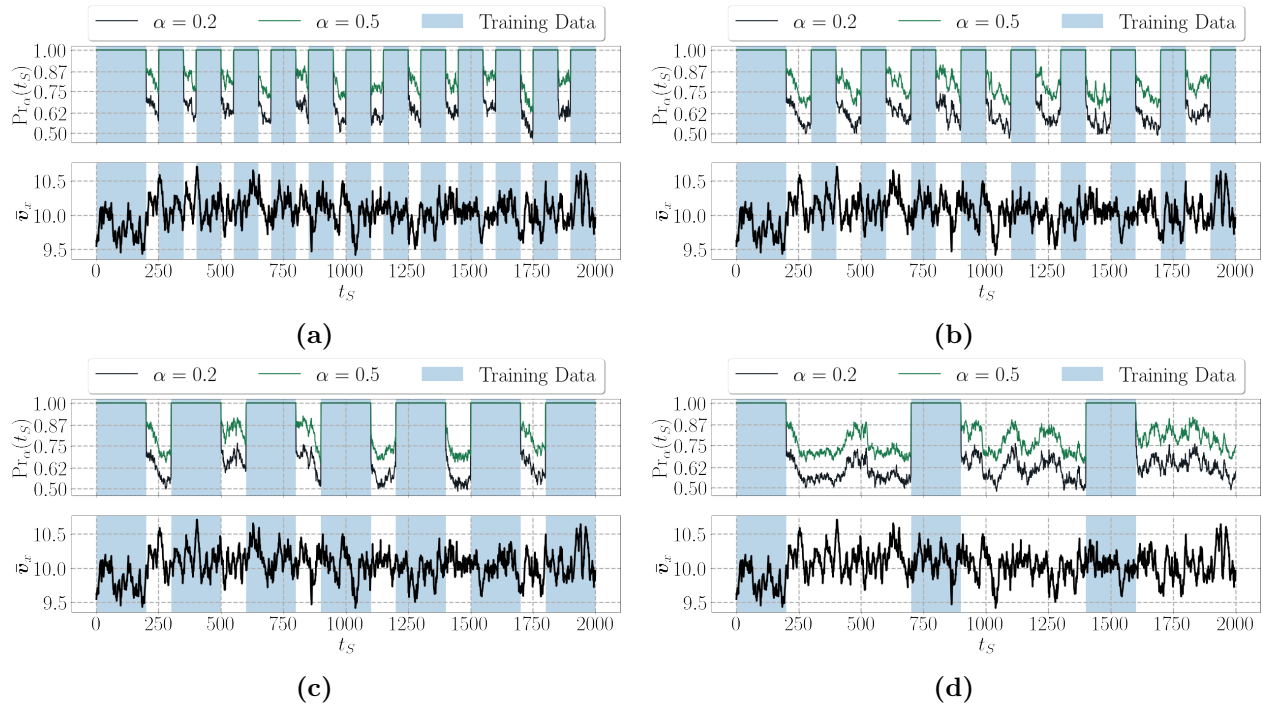


Figure 4.9: The top row shows the probability of achieving absolute vorticity prediction errors below thresholds of 0.2 (orange) and 0.5 (green) for the four test cases: T1 (a), T2 (b), T3 (c) and T4 (d), in the turbulent flow scenario. The bottom row depicts the temporal evolution of the spatially averaged streamwise velocity. These error thresholds are selected based on the typical vorticity range observed in the flow, as shown in Figure 4.8.

As shown in Fig. 4.9, the probability of achieving an absolute vorticity prediction error less than or equal to $\alpha = 0.5$ remains above 70% throughout most of the prediction horizon, with only occasional localized drops. This suggests that the POD-DL model's predictions generally

stay close to the reference dynamics and do not exhibit significant divergence. A similar trend is observed for $\alpha = 0.2$, where the probability consistently exceeds 50%. Considering that an absolute error of 0.2 corresponds to a relatively small error margin in the context of a turbulent flow, maintaining this level of accuracy indicates that the POD-DL model is effectively capturing the dominant features of the flow dynamics.

This conclusion is further supported by the comparison of TKE, computed as defined in eq. (4.5), between the ground truth and the predictions generated by the adaptive framework. This comparison, illustrated in Fig. 4.10, shows that the absolute TKE error remains small relative to the overall energy scale. The most pronounced discrepancy appears in test case T4, which involves the largest number of predicted time steps, reflecting the increasing challenge of maintaining long-term accuracy in turbulent regimes.

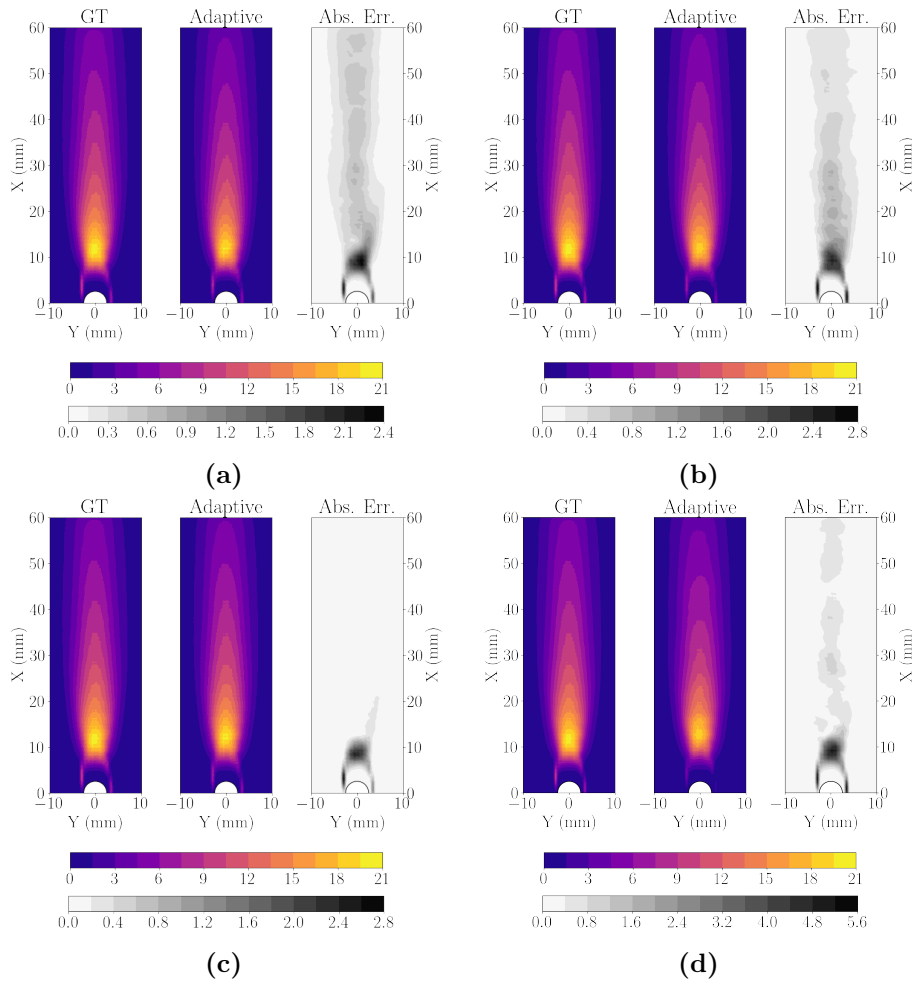


Figure 4.10: Turbulent kinetic energy (TKE) computed from the ground truth (left) and the adaptive framework predictions (middle) for test cases T1 (a), T2 (b), T3 (c) and T4 (d) in the turbulent flow scenario. The right panels display the absolute prediction error in TKE for each case.

The consistency of predictions with the underlying flow physics is further supported by visual inspection of representative snapshots comparing predicted and reference flow fields, as shown

in Fig. 4.8. In addition, Fig. 4.11 presents a comparison of the 25th, 50th (median) and 75th percentiles of the vorticity field, computed for each snapshot. The predicted percentiles closely align with those of the ground truth, reflecting the model’s ability to capture the overall statistical behavior of the flow. This is particularly significant in the context of highly complex, turbulent flows, where achieving exact pointwise agreement is often unrealistic.

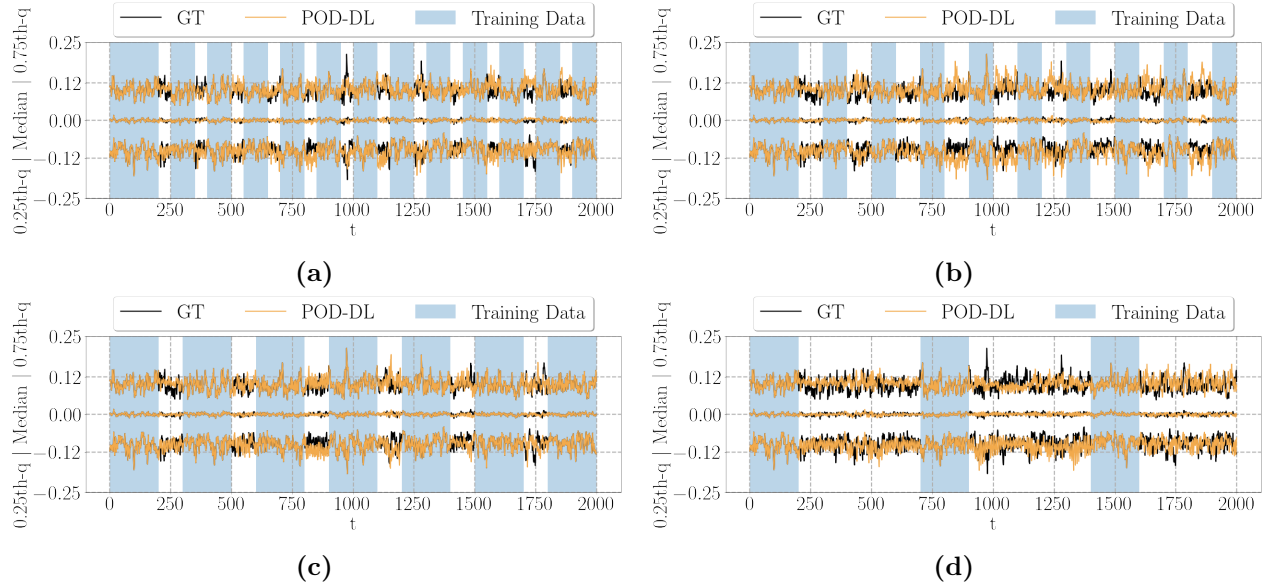


Figure 4.11: Temporal evolution of the 0.25-th (bottom), 0.5-th (middle) and 0.75-th (top) vorticity percentiles for the ground truth (blue) and POD-DL predictions (orange) across test cases T1 (a), T2 (b), T3 (c) and T4 (d) in the turbulent flow scenario. Shaded regions denote the time intervals corresponding to snapshots used during the initial and retraining phases.

Across all test cases examined for the turbulent flow scenario, the adaptive framework demonstrates the ability to adapt effectively to variations in complex flow dynamics. This adaptability is achieved through the incorporation of new ground truth data and the use of transfer learning, wherein the weights from prior training iterations are leveraged to enhance subsequent retraining phases.

Training the POD-DL model using 13 retained POD modes and a dataset of 200 snapshots requires approximately 1.76 minutes on an Intel(R) Core(TM) i7-10700 CPU @ 2.90GHz with 16 cores and 64 GB of RAM. As in the laminar case, the time required for generating predictions is nearly negligible, averaging around 2 seconds using the same hardware configuration. Based on these computational characteristics, the theoretical computational savings, quantified by the TCS metric defined in eq. (4.3), vary across test cases. Specifically, average savings of approximately 30% are achieved in test cases T1 and T3, 45% in T2 and up to 70% in T4.

4.4.3 Isothermal subsonic jet

In contrast to the previous two cases, which involved flow past a circular cylinder at different Reynolds numbers, this section evaluates the adaptive framework using a dataset derived from

a numerical simulation of an isothermal subsonic jet Brès et al., 2018. The jet issues from a round nozzle with an exit diameter of $D = 50$ millimeters (mm) at $\text{Re} \approx 1 \times 10^6$, based on D and the nozzle-exit velocity. The computations were carried out with the finite-volume solver CHARLES, which integrates the spatially filtered compressible Navier–Stokes equations on an unstructured mesh of ≈ 16 million control volumes; a third-order Runge–Kutta scheme advances the solution in time. Sub-grid stresses and heat fluxes are closed with the Vreman eddy-viscosity model, and the simulation covers 2000 acoustic time units, tc_∞/D .

For post-processing, the instantaneous fields were interpolated onto a structured cylindrical grid that preserves the LES resolution in the region of interest: $x/D \in [0, 30]$ along the streamwise direction and $r/D \in [0, 6]$ radially. The present study retains the streamwise velocity component only, sampled every $\Delta t = 0.02c_\infty^{-1}D$, where every sample is distributed on a structured grid of 39×175 spatial points. This yields 2000 equispaced snapshots arranged as,

$$\hat{D} \in \mathbb{R}^{1 \times 39 \times 175 \times 1 \times 2000} \quad (4.8)$$

As in the turbulent cylinder wake case, the complexity of the subsonic jet flow led to the exploration of various hyperparameter configurations within the adaptive framework. These test cases are summarized in Table 4.5. To maintain consistency with previous experiments, the number of snapshots used for initial training was kept relatively low compared to the total dataset. However, given the relatively simpler dynamics of this jet configuration, the initial number of training snapshots was reduced further to $S_0 = 100$.

Table 4.5: Hyperparameters used to test the adaptive framework on the isothermal subsonic jet.

Test	P	S_0	S_1	E
$T1$	100	100	50	1500
$T2$	200	100	100	1500
$T3$	600	100	100	1500
$T4$	2000	100	100	1500

As in the previous scenarios, POD is first applied to the initial $S_0 = 100$ snapshots to compute the corresponding POD coefficients. The number of retained modes is selected based on the proportion of energy contained in the reduced-order representation, using the cumulative energy function defined in eq. (2.25). In line with previous studies A. Martínez-Sánchez et al., 2023, retaining approximately 30% of the total energy is typically sufficient to capture the dominant dynamics in flows of this nature. As shown in Fig. 4.12, this energy threshold is achieved by retaining the first 5 POD modes from the initial set of snapshots for the isothermal jet.

This truncation to 5 modes is consistently applied across all training iterations within the adaptive framework to maintain architectural consistency in the POD-DL model. Unlike the previous cases, this dataset contains only the streamwise velocity component, hence, the model is trained and evaluated directly on the velocity field rather than derived quantities such as vorticity.

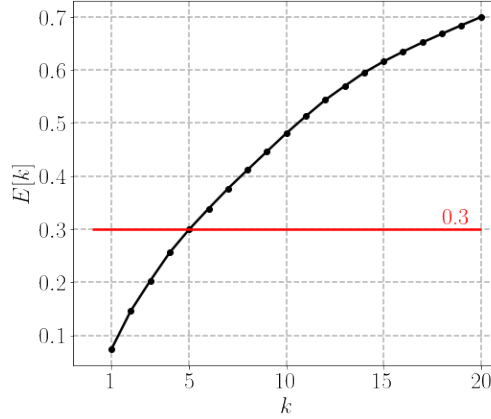


Figure 4.12: Percentage of energy contained, $E(k)$, when the truncation is done at the n , up to 20, most energetic modes in the isothermal jet.

As with the other flow scenarios, Fig. 4.13 provides a visual comparison between predicted and ground truth velocity fields at selected time steps for the isothermal jet. It is evident from this figure that the range of velocity values in this case is considerably narrower than in the other datasets, approximately spanning the interval $[4.405, 4.495]$. Based on this range, an absolute prediction error of 0.02 corresponds to a relative error in the range $[0.44\%, 0.45\%]$, while an error of 0.01 yields a relative error in the range $[0.22\%, 0.23\%]$. These magnitudes are notably small, suggesting that predictions achieving such levels of accuracy are highly reliable.

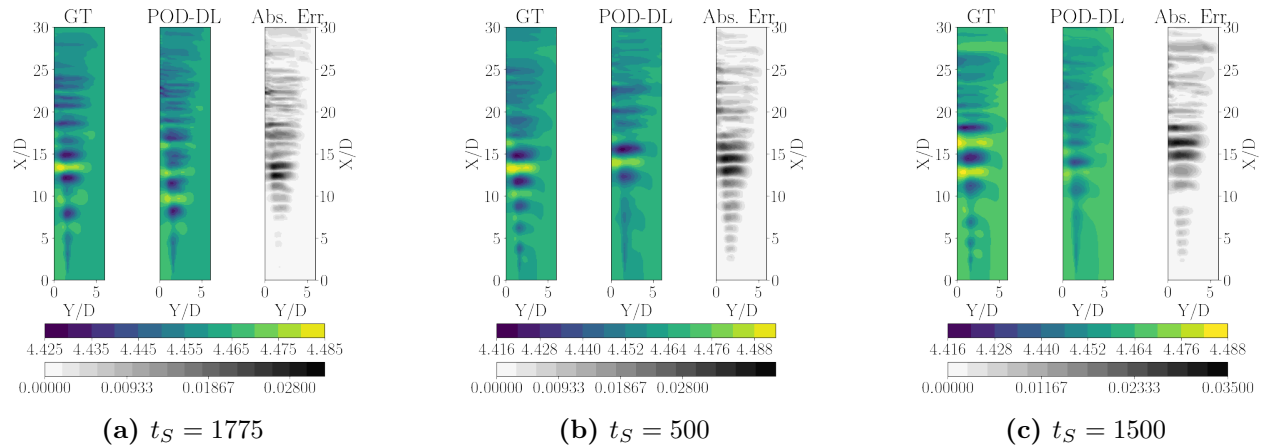


Figure 4.13: Snapshots comparing ground truth (GT) and POD-DL predicted velocity fields for the isothermal subsonic jet at representative time steps from test cases T1 (a), T3 (b) and T4 (c). The index t_S denotes the snapshot number, not the physical time t .

These probabilities are quantified in Fig. 4.14, which presents the probability of obtaining an absolute prediction error less than or equal to 0.01 and 0.02 across the four test cases defined in Tab. 4.5. This probability is computed using the PPE metric defined in eq. 4.2, expressed as $\text{Pr}\alpha(t_S) \equiv \Pr(-\alpha \leq \mathbf{v}_s - \hat{\mathbf{v}}_s \leq \alpha)$ for $\alpha \in \{0.01, 0.02\}$, where $\mathbf{v}_s$ and $\hat{\mathbf{v}}_s$ represent the s -th ground truth and predicted velocity snapshots, respectively.

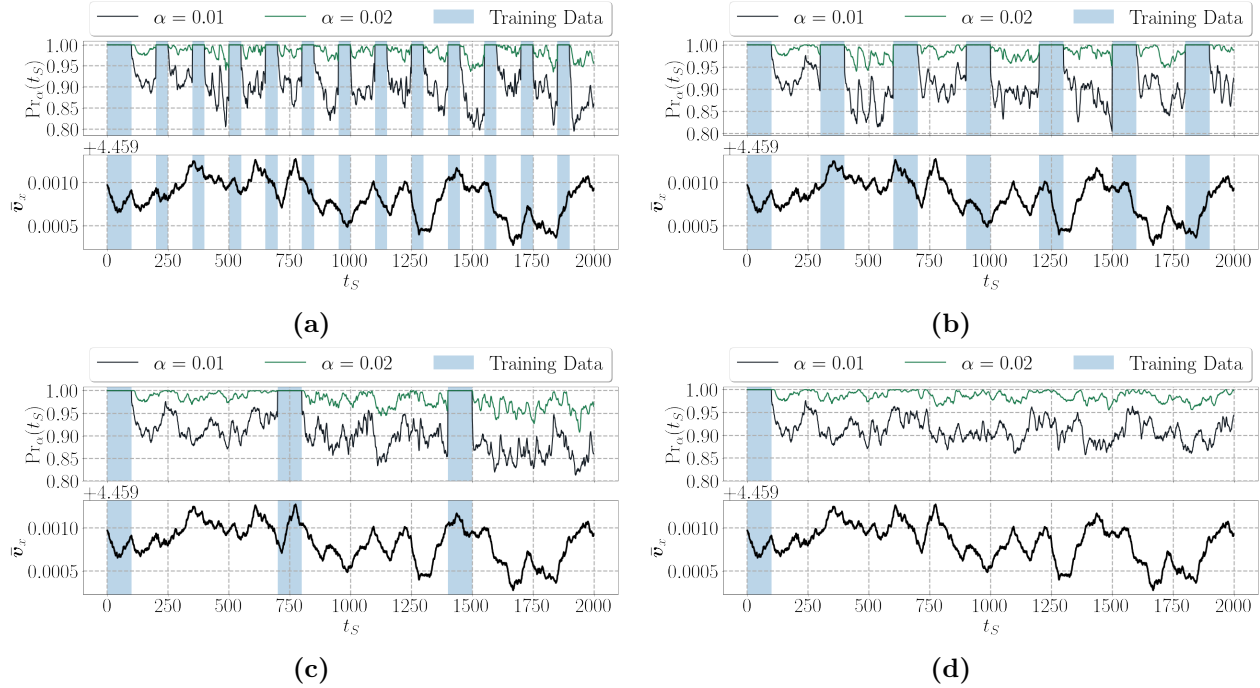


Figure 4.14: The top row shows the probability of achieving absolute velocity prediction errors below thresholds of 0.01 (orange) and 0.02 (green) for the four test cases: T1 (a), T2 (b), T3 (c) and T4 (d), in the isothermal jet scenario. The bottom row depicts the temporal evolution of the spatially averaged streamwise velocity. These error thresholds are selected based on the typical velocity range observed in the flow, as shown in Fig. 4.13.

As shown in Fig. 4.13, the range of velocity values in the isothermal jet is considerably narrower than in the other flow scenarios, approximately spanning the interval $[4.405, 4.495]$. Consequently, an absolute prediction error of 0.02 in the velocity field is negligible for this flow. A model that consistently achieves a high probability (i.e., $Pr_\alpha(t_s) \simeq 1$) of maintaining prediction errors below this threshold can be considered highly accurate. As illustrated in Fig. 4.14, this condition is generally satisfied across all predictions produced by the POD-DL model, with deviations occurring only at isolated time steps.

The same figure also presents the probability of achieving an absolute error less than or equal to 0.01, a threshold that is even more stricter. Remarkably, this probability remains above 80% throughout the entire prediction horizon, indicating that the POD-DL model maintains close alignment with the reference dynamics and does not exhibit significant divergence over time.

These results can largely be attributed to the statistically steady nature of the isothermal jet flow, which has reached a statistically converged state with minimal temporal variability. This is evident in the nearly flat evolution of the spatially averaged streamwise velocity, shown in the bottom row of all panels in Fig. 4.14, where fluctuations are negligible. Such dynamical stability allows the POD-DL model to capture the dominant flow behavior effectively, even when trained on a relatively small dataset.

Further insight is provided by Fig. 4.15, which compares the 0.25-th, 0.5-th (median) and 0.75-th velocity percentiles derived from the ground truth and the adaptive framework predictions. The variability in these percentile values is minimal, and although the POD-DL model generally follows the trend, it shows limited capacity to resolve subtle fluctuations. Notably, in test cases T1 and T2, where more training data are incorporated, the model exhibits improved alignment with these fine-scale variations. This suggests that iterative retraining with new data enhances the model’s ability to adapt to even small perturbations in the flow.

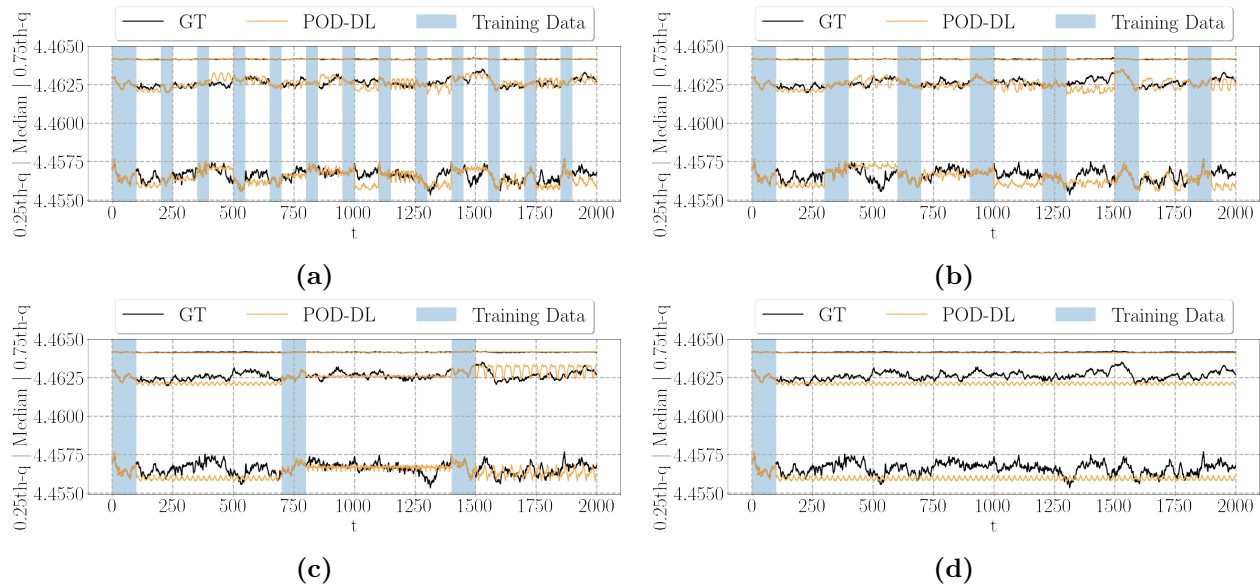


Figure 4.15: Temporal evolution of the 0.25-th (bottom), 0.5-th (middle) and 0.75-th (top) velocity percentiles for the ground truth (blue) and POD-DL predictions (orange) across test cases T1 (a), T2 (b), T3 (c) and T4 (d) in the isothermal jet scenario. Shaded regions indicate the snapshot intervals used during both initial training and retraining phases.

Training the POD-DL model with 5 retained POD modes and a training set of 100 snapshots requires approximately 1.18 minutes on an Intel(R) Core(TM) i7-10700 CPU @ 2.90GHz with 16 cores and 64 GB of RAM. As with the other flow scenarios, the time required for generating predictions is nearly negligible, averaging around 2 seconds on the same hardware. Under these conditions, the theoretical computational savings, quantified using the TCS metric defined in eq. (4.3), vary across test cases: approximately 65% for T1 and T2, around 85% for T3 and up to 95% for T4. These findings highlight the effectiveness of the adaptive framework in substantially reducing the computational cost associated with traditional numerical solvers, while maintaining a high level of predictive fidelity.

The following section provides a general discussion of the results obtained with the adaptive framework and outlines potential directions for future research.

4.5 Discussion

The adaptive framework proposed in this study was evaluated across three datasets, each representing flow regimes with varying Reynolds numbers. As discussed in Section 4.4, the highest prediction accuracy was achieved for the laminar flow past a circular cylinder (Section 4.4.1) and the isothermal subsonic jet (Section 4.4.3). The superior performance of the POD-DL model for the isothermal jet is attributed to the flow’s statistically steady regime, which ensures a stable data distribution over time. This consistency is clearly reflected in the bottom panels of Fig. 4.14, which show negligible variation in the spatially averaged streamwise velocity.

Because the temporal correlations in the data remain stable, a sufficiently large training dataset allows the model to learn these patterns effectively and extrapolate the evolution of the POD coefficients with high accuracy.

In contrast, for the laminar cylinder flow, when the data distribution changes, the model’s predictions begin to diverge due to a lack of information of the new dynamics. This divergence is evident in Fig. 4.5 for test cases T1 (a) and T3 (c), where the model fails to capture a shift in the dynamics.

To address this challenge, the adaptive framework incorporates periodic retraining of the POD-DL model using newly acquired data. This allows the model to account for evolving dynamics. The benefit of this strategy is demonstrated in Fig. 4.5 (b) for test case T2, where retraining enables long-term stability and accuracy in the predictions.

A closer look at Fig. 4.5 (a) reveals that, even after a second retraining, the model continues to diverge. This occurs because the new training data did not sufficiently capture the upcoming change in dynamics. However, after a third retraining on data from a more stable state, the predictions return to alignment with the reference. This example illustrates the adaptability and resilience of the framework in the face of evolving system dynamics.

These findings highlight the importance of an adaptive mechanism to accommodate shifts in the data distribution, an essential capability for handling complex fluid dynamics problems.

The most challenging case analyzed involved the turbulent three-dimensional wake behind a circular cylinder (Section 4.4.2), which exhibited frequent and significant variations in the spatial mean velocity (Fig. 4.9, bottom row). These fluctuations, caused by small-scale nearly-chaotic structures, limited the horizon over which the POD-DL model could maintain accurate predictions. Despite this, retraining with new data allowed the model to recover accuracy, as seen in the transient improvements following each retraining phase.

This behavior is further supported by Fig. 4.10, which compares the turbulent kinetic energy (TKE) of the reference and predicted dynamics. Although prediction accuracy degrades over time, retraining consistently restores convergence, demonstrating the effectiveness of the adaptive framework.

Nonetheless, one of the main limitations lies in the forecasting model itself. The current POD-DL architecture, while computationally efficient and interpretable, shows limited extrapolation capability in complex regimes and sensitivity to distributional shifts. Future work may explore

alternative DL models with enhanced temporal representation capabilities and robustness. Promising candidates include extended long short-term memory (xLSTM) networks (Beck et al., 2024), multi-layer perceptron (MLP) mixer models (Ekambaram et al., 2023) and diffusion models (Lin et al., 2024b).

In particular, diffusion models have attracted attention for their ability to approximate complex probability distributions, also there is a growing interest in the literature in evaluating their ability to capture extreme events within datasets (Stamatelopoulos and Sapsis, 2025). This capability is relevant for forecasting previously unseen behaviors in turbulent flows. However, their integration into the proposed adaptive framework remains a subject for future exploration.

Another area left for future investigation is uncertainty quantification (UQ). A robust UQ strategy would enable dynamic truncation of predictions based on estimated reliability, without requiring access to the ground truth. Several methods may be adapted for this purpose, including bootstrap ensembles (Kičić et al., 2023; Lakshminarayanan et al., 2017), POD-based structural approaches (Abadía-Heredia, Lopez-Martín, and le Clainche, 2024; Beltrán et al., 2022; Rapún et al., 2015), conformal prediction (Auer et al., 2023) and residual-based evaluations of the governing PDEs (Rapún et al., 2015).

Finally, while this chapter used a precomputed dataset as a surrogate for the numerical solver, future work will integrate a live numerical solver into the adaptive loop. Although invoking a solver at retraining intervals may reduce computational savings, this cost is expected to be modest given that solvers generally require only a few time steps to re-stabilize after POD-DL prediction. Therefore, the trade-off between accuracy and efficiency could still be favorable.

Overall, this work provides a proof of concept for an adaptive DL-based framework for accelerating CFD simulations. It demonstrates both the potential for significant computational savings and the need for further development of model architectures, UQ strategies and solver integration to fully realize its capabilities.

Chapter 5

Generative AI for capturing small-scale structures in turbulence

As demonstrated in preceding chapters, DL-based surrogate models offer a significant speed advantage over conventional CFD solvers. However, they often underperform in accurately capturing the small-scale structures embedded within the flow dynamics. In the case of the POD-DL model, this limitation arises primarily because these fine-scale features are typically associated with low-energy POD modes, whose temporal coefficients exhibit highly irregular, noise-like behavior that challenges data-driven prediction.

For fully data-driven models, such as the residual and variational autoencoders discussed in Chapter 3, the difficulty is even more pronounced. Unlike POD-DL frameworks, which operate on temporally matrix-format inputs, these models ingest spatio-temporal tensors consisting of high-dimensional snapshots. Consequently, successful reconstruction of small-scale structures requires the model to both spatially localize these features and capture their temporal evolution, a substantially more complex task.

A growing body of literature suggests that the challenge of modeling small-scale flow structures can be addressed through the use of more advanced deep learning architectures (Du et al., 2024; Fotiadis et al., 2024; Gao, Kaltenbach, and Koumoutsakos, 2024; Shysheya et al., 2024). These approaches emphasize the inherently stochastic nature of fine-scale dynamics, advocating for models that not only predict deterministic outcomes but also approximate the underlying probability distributions from which such structures arise.

Deep learning frameworks capable of modeling such distributions and sampling from them are identified in the literature as generative models. While the VAE examined in Chapter 3 constitutes one example of this class, recent advances in diffusion-based models have demonstrated superior generative performance relative to VAEs and GANs (Dhariwal and Nichol, 2021; Ho et al., 2020; Song et al., 2021).

This chapter extends the framework introduced by Gao, Kaltenbach, and Koumoutsakos, 2024, who proposed a generative model termed Generative Learning of Effective Dynamics (G-LED). This model integrates a Transformer architecture (Vaswani et al., 2017) with a denoising diffusion probabilistic model (DDPM) (Ho et al., 2020), enabling it to forecast both

large- and small-scale structures in high-dimensional spatio-temporal turbulent flows.

Here, G-LED is trained and evaluated on the turbulent flow past a circular cylinder (Mendez et al., 2019), used in previous chapters, demonstrating its capacity to capture and forecast fine-scale dynamics. Furthermore, we explore the incorporation of G-LED within the POD-DL framework. The following section provides a detailed description of the G-LED architecture and its integration with the POD-DL methodology.

5.1 Method: G-LED

This chapter focuses on using generative models for forecasting high-dimensional spatio-temporal dynamics, which are usually described by a system of non-linear partial differential equations (PDEs). There are generally two ways of obtaining the dynamics corresponding to the physical system: by experimental measurements or by calculating the solution of the PDEs system. The former is usually done following the technique of particle image velocimetry (PIV), which requires specialize and expensive hardware, however experimental studies offers information about dynamics that are highly difficult to simulate.

Conversely, numerical methods (Tadmor, 2012) are employed to approximate solutions to the governing equations (2.1), as analytical solutions are generally intractable. In this approach, the system of PDEs is discretized in both time and space. Let $\mathbf{v}t \in \mathbb{R}^{C \times N}$ denotes the discretized solution at time t , where C represents the number of components, such as velocity components in the streamwise, wall-normal and spanwise directions, and N denotes the number of spatial discretization points. Note, N depends on the spatial dimensionality of the problem, for instance, in two-dimensional cases $N = N_x \times N_y$, where N_x and N_y are the number of grid points along the x - and y -axes, respectively. The solution at the next time step is computed based on the preceding temporal state according to,

$$\mathbf{v}_{t+\delta t} = \mathcal{F}(\mathbf{v}_t). \quad (5.1)$$

Here $\mathcal{F} : \mathbb{R}^{C \times N} \longrightarrow \mathbb{R}^{C \times N}$ denotes the numerical operator that advances the solution by a predefined time step δt . This iterative process requires an initial condition, $\mathbf{v}_0$. After propagating the solution over S time steps, the resulting dataset, $\mathring{\mathbf{D}}$, is given by

$$\mathring{\mathbf{D}} = [\mathbf{v}_0, \dots, \mathbf{v}_{S-1}], \quad \mathring{\mathbf{D}} \in \mathbb{R}^{C \times N \times S}, \quad (5.2)$$

where the solution, $\mathbf{v}_t$, at time t is referred to as a snapshot, as it typically corresponds to a two- or three-dimensional volumetric field capturing the system's spatio-temporal dynamics. This terminology equally applies to data acquired from experimental measurements, through PIV, where snapshots represent flow fields reconstructed from post-processed image sequences captured by a high-speed camera, and stored in a dataset like in eq. (5.2).

In practical scenarios, $N \gg 1$, leading to a substantial number of degrees of freedom, often on the order of thousands to millions, which significantly increases the computational cost

and makes performing DNS of the governing equations computationally intensive and time-consuming.

Following the work from Gao, Kaltenbach, and Koumoutsakos, 2024, it is assumed that, over short temporal horizons, the system state at time t , denoted $\mathbf{v}_t$, can be effectively approximated by a lower-dimensional representation $\mathbf{z}_t \in \mathbb{R}^{C \times N_z}$, where $N_z \ll N$. Throughout this chapter, $\mathbf{v}_t$ and $\mathbf{z}_t$ will be referred to as the micro and macro representations of the solution at time t , respectively.

The macro representation can be derived using various types of encoders (Baldi, 2021), which take the micro solution as input. However, to minimize the introduction of unnecessary trainable parameters, a non-trainable encoder $\mathcal{E} : \mathbb{R}^{C \times N} \rightarrow \mathbb{R}^{C \times N_z}$, which in this case is a multi-linear downsampler, is employed to map the micro to the macro representation, as follows,

$$\mathbf{z}_t = \mathcal{E}(\mathbf{v}_t), \quad (5.3)$$

This encoder operates exclusively along the spatial dimension, preserving the number of components C .

5.1.1 Forecasting model: Transformer

The macro representation $\mathbf{z}_t$ is characterized by the absence of small-scale structures, which are typically associated with near-chaotic dynamics. By retaining only the dominant large-scale features, the resulting dynamics become substantially more tractable for modeling. To forecast the temporal evolution of this reduced-order system, a macro propagator $\mathcal{A}^{\theta_A}$, parameterized by trainable weights θ_A and based on the Transformer architecture, is employed,

$$\mathcal{A}^{\theta_A} : \begin{cases} (\mathbf{z}_0, \mathbf{z}_{\Delta t}, \dots, \mathbf{z}_{(n_t-1)\Delta t}) \rightarrow \mathbf{z}_{n_t\Delta t}, & \text{if } n_t \leq N_t, \\ (\mathbf{z}_{(n_t-N_t)\Delta t}, \mathbf{z}_{(n_t-N_t+1)\Delta t}, \dots, \mathbf{z}_{(n_t-1)\Delta t}) \rightarrow \mathbf{z}_{n_t\Delta t}, & \text{if } n_t > N_t. \end{cases} \quad (5.4)$$

Here, N_t represents the maximum number of time steps the macro propagator can look back at. Following previous studies: Geneva and Zabaras, 2022; Sun et al., 2023. The optimal trainable weights θ_A^* of the propagator are obtained by minimizing the MSE,

$$\theta_A^* = \arg \min_{\theta_A} \sum_{n_t \in \dot{\mathbf{D}}_{train}} \sum_{i=1}^{N_t} \|\mathbf{z}_{n_t\Delta t} - \mathcal{A}^{\theta_A}(\mathbf{z}_{(n_t-i)\Delta t}, \dots, \mathbf{z}_{(n_t-1)\Delta t})\|_2^2. \quad (5.5)$$

Here, $\mathcal{D}_{train}$ denotes the subset of temporal snapshots used for training the propagator. The full dataset $\dot{\mathbf{D}}$ is partitioned into disjoint training and testing sets, such that $\dot{\mathbf{D}} = \dot{\mathbf{D}}_{train} \cup \dot{\mathbf{D}}_{test}$, where $\dot{\mathbf{D}}_{test}$ is reserved exclusively for model evaluation.

5.1.2 Recovering of micro dynamics: generative model

Following the prediction of future macro states, their corresponding micro representations are reconstructed using a DDPM. This generative framework maps a temporal window of macro inputs, $\mathbf{z}_{t:t+B\Delta t}$, to the corresponding micro outputs, $\mathbf{v}_{t:t+B\Delta t}$.

A DDPM is a generative framework composed of two sequential stages: a forward (noising) process and a reverse (denoising) process. The forward process progressively corrupts the micro data by adding Gaussian noise over N_e steps according to a Markov chain,

$$\mathbf{v}_{t:t+B\Delta t} \equiv \mathbf{e}^0 \rightarrow \mathbf{e}^1 \rightarrow \dots \rightarrow \mathbf{e}^{N_e}, \quad (5.6)$$

$$\mathbf{e}^i \sim \mathcal{N}\left(\mathbf{e}^i; \sqrt{1 - \sigma_i^2} \mathbf{e}^{i-1}, \sigma_i^2 \mathbf{I}\right), \quad i = 1, \dots, N_e, \quad (5.7)$$

where $\{\sigma_i^2\}_{i=1}^{N_e}$ is a pre-defined variance schedule. This schedule is strictly monotonically increasing (Ho, Salimans, et al., 2022) and it is chosen such that early noise levels are minimal, preserving the key features of the original data (i.e., $\mathbf{e}^0 \simeq \mathbf{e}^1$), while later stages result in a fully randomized sample (i.e., $\mathbf{e}^{N_e} \sim \mathcal{N}(0, \mathbf{I})$) with $\sigma_{N_e} \gg \|\mathbf{v}_{t:t+B\Delta t}\|_2$ (D. Kingma et al., 2021). This schedule of variances, is computed as follows,

$$\sigma_i = \left(\sigma_{N_e}^{\rho-1} + \frac{N_e - i}{N_e - 1} (\sigma_1^{\rho-1} - \sigma_{N_e}^{\rho-1}) \right)^\rho, \quad \text{for } i = 1, 2, \dots, N_e. \quad (5.8)$$

By following the work of Ho, Chan, et al., 2022 $\rho = 7$ to achieve a controlled, non-linear progression of the variance schedule. The variable N_e is a hyperparameter of the diffusion model selected based on previous works: De Bortoli et al., 2021; Gao, Han, et al., 2024; Ho, Salimans, et al., 2022.

On the other hand, the reverse process starts from a white noise sample $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ and it learns to gradually remove the noise until recovering the original micro dynamics $\mathbf{v}_{t:t+B\Delta t}$. The reverse process used for G-LED is known as the variance-exploding diffusion process (Song and Ermon, 2020), where the reverse process is tractable (Chan, 2025; D. Kingma et al., 2021),

$$p(\mathbf{e}^i | \mathbf{v}_{t:t+B\Delta t}, \mathbf{e}^{i+1}) = \mathcal{N}\left(\frac{\sigma_{i+1}^2 - \sigma_i^2}{\sigma_{i+1}^2} \mathbf{v}_{t:t+B\Delta t} + \frac{\sigma_i^2}{\sigma_{i+1}^2} \mathbf{e}^{i+1}, \frac{(\sigma_{i+1}^2 - \sigma_i^2) \sigma_i^2}{\sigma_{i+1}^2} \mathbf{I}\right). \quad (5.9)$$

However, the micro dynamics $\mathbf{v}_{t:t+B\Delta t}$ are unknown, and therefore they are substituted by a deep neural network (DNN), $\hat{\mathbf{S}}_{\theta_D} : \mathbb{R}^{C \times N_z \times B} \times \mathbb{R}^{C \times N \times B} \times \mathbb{N} \rightarrow \mathbb{R}^{C \times N \times B}$, parameterized by trainable weights θ_D . This DNN receive as input the macro dynamics, and it is tasked to approximate the micro dynamics. The reverse process is therefore expressed as follows,

$$p(\mathbf{e}^i | \mathbf{v}_{t:t+B\Delta t}, \mathbf{e}^{i+1}) \approx p(\mathbf{e}^i | \mathbf{z}_{t:t+B\Delta t}, \mathbf{e}^{i+1}, i) = \mathcal{N}\left(\frac{\sigma_{i+1}^2 - \sigma_i^2}{\sigma_{i+1}^2} \mathcal{D}^{\theta_D}(\mathbf{z}_{t:t+B\Delta t}, \mathbf{e}^{i+1}, i) + \frac{\sigma_i^2}{\sigma_{i+1}^2} \mathbf{e}^{i+1}, \frac{(\sigma_{i+1}^2 - \sigma_i^2)\sigma_i^2}{\sigma_{i+1}^2} \mathbf{I}\right). \quad (5.10)$$

The DNN employed within G-LED is a U-Net architecture (Ronneberger et al., 2015). The optimal parameters of the diffusion model, θ_D^* , are learned by minimizing the Kullback–Leibler (KL) divergence between the true and approximate reverse diffusion processes, denoted in eqs. (5.9) and (5.10). Under standard assumptions, as detailed by Chan, 2025; Ho et al., 2020, this minimization yields the following loss function:

$$\theta_D^* = \arg \min_{\theta_D} \sum_{n_t \in \hat{\mathcal{D}}_{train}} \mathbb{E}_{i, \mathbf{e}^i} \|\mathcal{D}^{\theta_D}(\mathbf{z}_{t:t+B\Delta t}, \mathbf{v}_{t:t+B\Delta t}^i, i) - \mathbf{v}_{t:t+B\Delta t}\|_2, \quad (5.11)$$

where $\hat{\mathcal{S}}_{\theta_D}$ denotes the network’s estimate of the noise at diffusion step i , conditioned on the macro inputs. The G-LED model results from coupling the macro propagator with the diffusion decoder, as originally proposed by Gao, Kaltenbach, and Koumoutsakos, 2024, for forecasting high-dimensional spatio-temporal dynamics. A schematic illustration of the complete methodology is provided in Fig. 5.1.

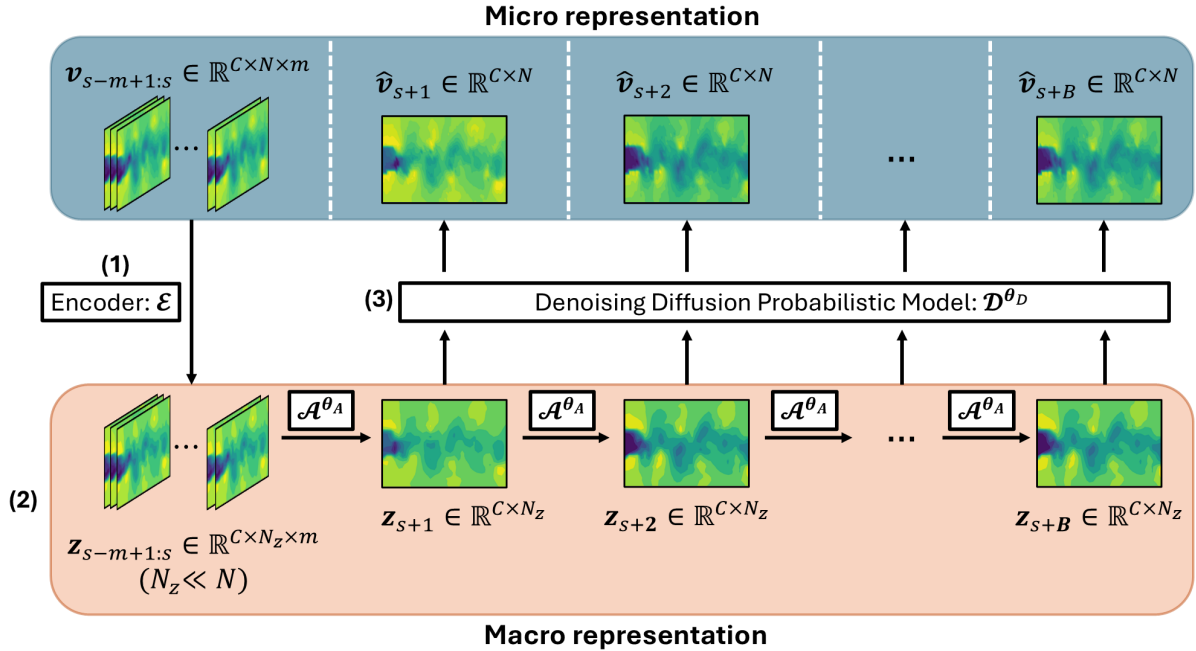


Figure 5.1: Overview of the G-LED methodology: (1) high-resolution images (micro) are first downsampled via an encoder, (2) the resulting low-resolution (macro) dynamics are then forecasted using a transformer architecture and (3) the micro dynamics are subsequently reconstructed by applying a diffusion-based model.

5.2 Implementation of G-LED within the POD-DL framework

This methodology was originally proposed for dynamics of diverse dimensionalities, even for one-dimensional problems, like the 1D viscous Burgers' equation. This is achieved by modifying the dimensionality of the convolutional kernel within the U-Net. In this way G-LED was applied to the 1D Kuramoto-Sivashinsky equation, a 2D flow over backward-facing step and a 3D turbulent channel flow in the work of Gao, Kaltenbach, and Koumoutsakos, 2024.

As it has been shown throughout this thesis the implementation of POD in DL frameworks allows to decrease the dimensionality of a problem from a high-dimensional tensor dataset, $\hat{\mathbf{D}}$, to a low-dimensional matrix, $\mathbf{\Gamma}$, composed by vectors representing the POD coefficients as shown in (2.10).

In this way G-LED can be integrated in the POD-DL methodology, where in previous chapters the DL model was composed by an LSTM and Dense layers, now it can be substituted by G-LED. This biggest disadvantage of previous models was the number of retained modes, since a large number of modes may negatively affect the performance of training, and therefore the accuracy of predictions. As explained at the beginning of this chapter, this is due to the noisy dynamics corresponding to low energetic modes. However, the ability of G-LED to capture highly-complex distributions is a valuable capability for this kind of problems. Figure 5.2 shows a visual depiction of how G-LED is integrated within the POD-DL methodology. The following section shows the results obtained from G-LED and its implementation on the POD-DL methodology.

5.3 Results

The work described in this chapter is the continuation of the work realized by Gao, Kaltenbach, and Koumoutsakos, 2024, where the generative model G-LED was presented and evaluated in forecasting different spatio-temporal dynamics, all of them obtained from numerical simulations and representing bed problems of fluid dynamics.

The objective of this chapter is to evaluate the performance of G-LED using the dataset from the 3 kHz time-resolved PIV sequence measurements of Mendez et al., 2020, obtained behind a 5 mm circular cylinder in the L10 low-speed tunnel at the von Kármán Institute. At the start of the record, using a Optronis SpeedSense 9090 CMOS camera, the free-stream velocity is $U_\infty = 12.1$ m/s ($\text{Re} \approx 4 \times 10^3$), after 1 second it is ramped down to 7.9 m/s ($\text{Re} \approx 2.6 \times 10^3$) and then held constant. The complete run therefore contains two statistically stationary regimes separated by a controlled transient state. A total of 13200 frames were processed with the DYNAMICSTUDIO software, giving planar velocity fields on a 301×111 vector grid every $\Delta t = 0.333$ ms. For convenience we refer to the three contiguous blocks as "high-Re steady" ($s = 1 : 4000$), "transient" ($s = 4001 : 8000$) and "low-Re steady" ($s = 8001 : 13200$).

In contrast to previous chapters, where deep learning models were trained using data from a single steady-state regime, G-LED is here trained on a combined subset of snapshots from both steady-states (low- and high-Re). Specifically, the first 3200 snapshots from each regime

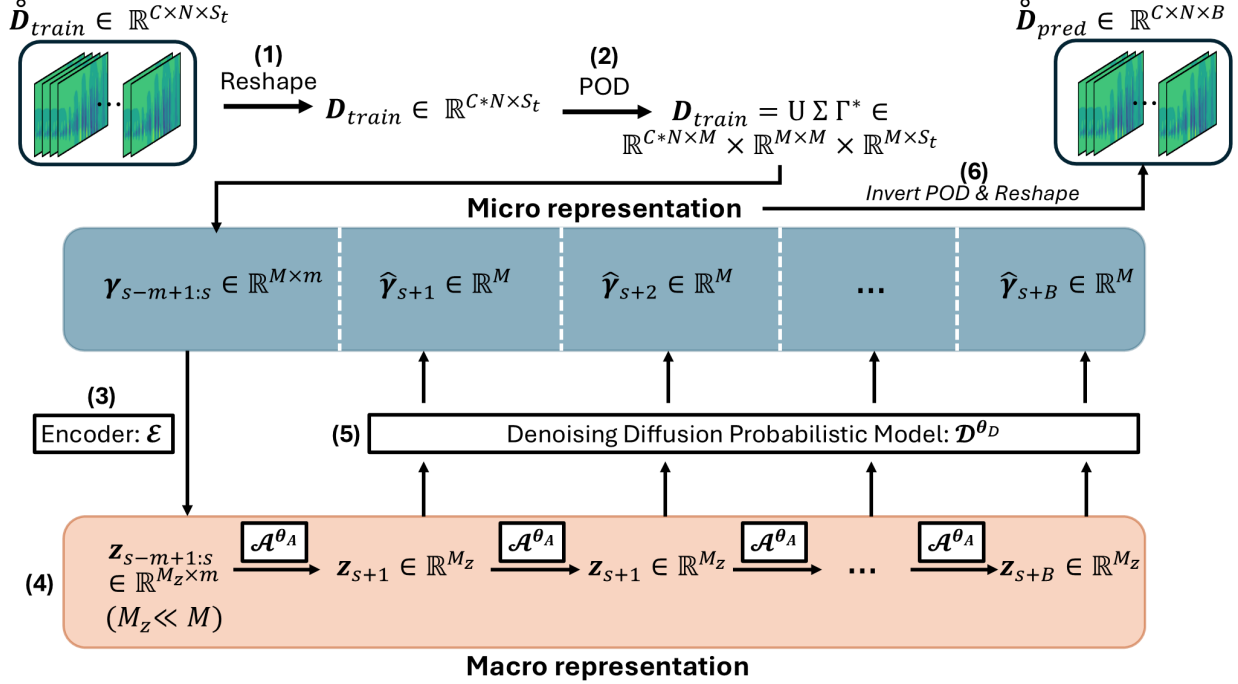


Figure 5.2: Schematic illustration of the implementation of G-LED within the POD-DL methodology: As detailed in Chapter 2, (1) the original dataset, initially in tensor format, is reshaped into a matrix format, (2) POD is then applied to decompose this matrix into POD modes and coefficients, as described in eq. (2.10), enabling the retention of a selected number of POD modes and coefficients. (3) The number of retained POD coefficients is further reduced by the encoder, simplifying the dynamics to be predicted. (4) The reduced set of POD coefficients is then forecasted using the transformer model. (5) The diffusion model is employed to predict the POD coefficients that were discarded by the encoder. (6) Finally, the predicted snapshots are reconstructed by inverting the POD transformation using the original POD modes and the predicted POD coefficients.

are employed for training, resulting in a total of 6400 training samples. The model is then evaluated on its ability to forecast the subsequent 100 snapshots. Notably, this constitutes a relatively limited training set for a generative model.

5.3.1 Results from G-LED without POD

The micro dynamics in this dataset are represented as tensors of dimension $\mathbf{v}_t \in \mathbb{R}^{2 \times 111 \times 301}$. To accommodate the architectural requirements of the U-Net used in G-LED, specifically that the spatial dimensions must be multiples of 16 due to the convolutional kernel, the data are linearly interpolated to a compatible resolution, resulting in $\mathbf{v}_t \in \mathbb{R}^{2 \times 128 \times 320}$.

The macro representation targeted by the encoder is $\mathbf{z}_t \in \mathbb{R}^{2 \times 32 \times 32}$, effectively filtering out small-scale features and thus simplifying the modeling task for the macro propagator. The removed fine-scale dynamics are subsequently reconstructed through the diffusion model. Following previous chapters, G-LED is trained on the velocity field, $\mathbf{v}$, but the vorticity field $\boldsymbol{\omega}$ is used to quantify the predictions and it is defined as in eq. (2.32).

To assess the prediction accuracy, the coefficient of determination (R^2) metric is employed. This metric evaluates the proportion of variance in the ground truth explained by the model prediction and is defined as,

$$R^2 = 1 - \frac{\sum_i (\omega_i - \hat{\omega}_i)^2}{\sum_i (\omega_i - \bar{\omega})^2}, \quad (5.12)$$

where ω_i denotes the true vorticity values, $\hat{\omega}_i$ the predicted values, and $\bar{\omega}$ the mean of the true values. An R^2 value close to 1 indicates a high level of predictive accuracy. This is shown in Fig. 5.3.

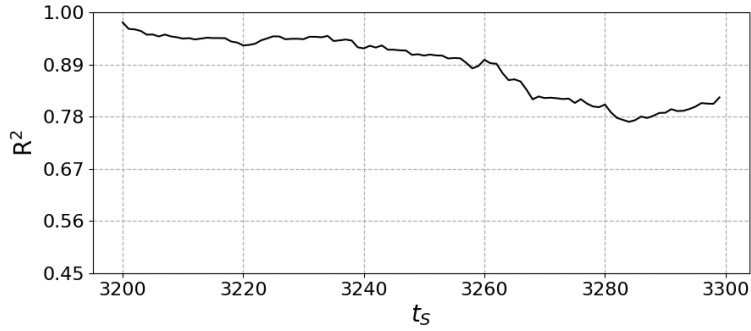


Figure 5.3: Value of the R^2 metric, defined in eq. (5.12), for each prediction generated by G-LED.

Figure 5.4 presents a comparison between G-LED predictions and ground truth data for both the mean vorticity and the root mean square (RMS) vorticity profile. The latter is defined as,

$$\overline{\omega(x)} \equiv \sqrt{\frac{1}{L_y S} \int_0^{L_y} \int_0^S \omega(x, y, t) dy dt}. \quad (5.13)$$

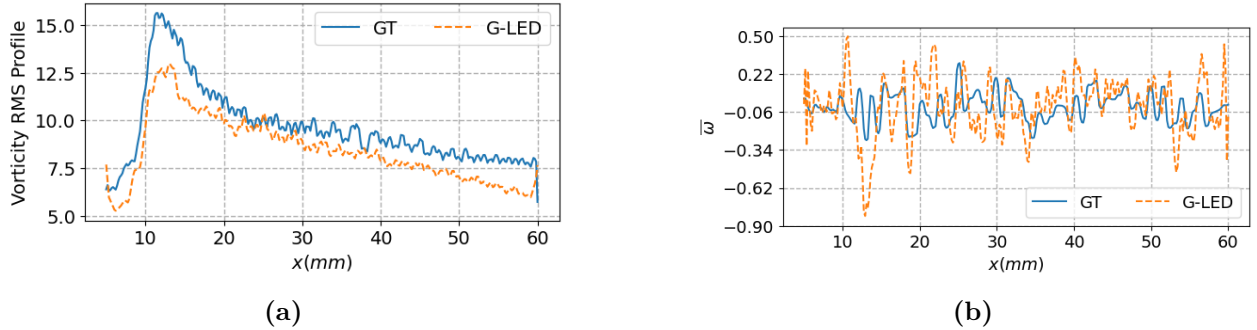


Figure 5.4: Comparison of the GT (solid line) and G-LED predictions (dashed lines) for the root mean square (RMS) vorticity profile along the streamwise direction (a) and the mean vorticity alongside the streamwise direction (b).

where L_y denotes the spatial domain length in the spanwise direction (y -axis) and S is the total number of predicted snapshots. Note, the RMS profile captures the spatiotemporal fluctuations

attributable to the small-scale structures, highlighting how G-LED is able to approximate these dynamics. This is also illustrated in Fig. 5.5 that displays representative vorticity snapshots generated by G-LED alongside those obtained from experimental measurements.

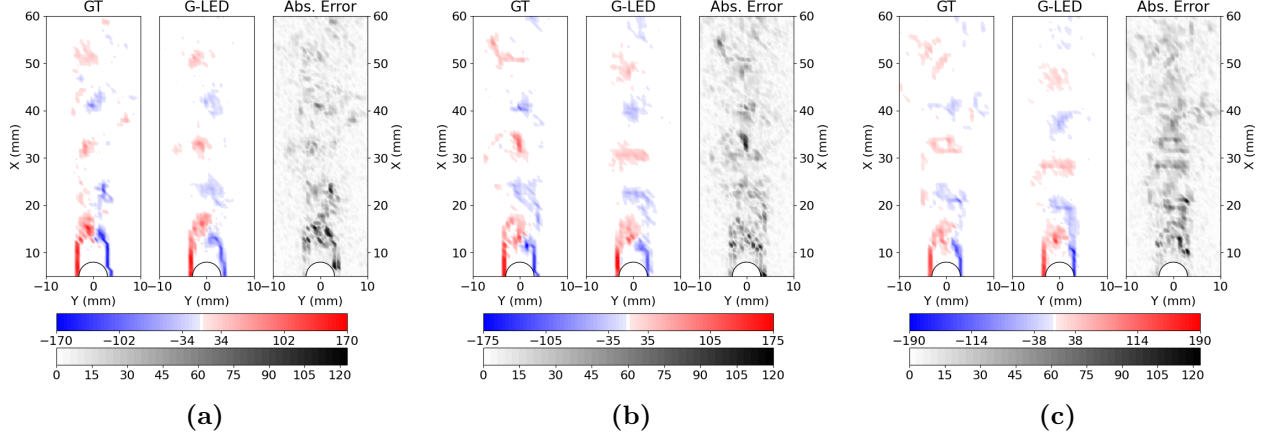


Figure 5.5: Comparison between the ground truth (GT) and predicted vorticity snapshots for snapshots (a) 3210, (b) 3250 and (c) 3300 corresponding to the second steady-state in the flow.

These results demonstrate that the generative model is capable of producing predictions that faithfully capture the dynamics associated with small-scale turbulent structures—features that are notoriously challenging to forecast with conventional data-driven approaches. Figure 5.6 presents a comparison between the predicted and ground truth turbulent kinetic energy (TKE), as defined in eq. (4.5). The results indicate that the generative model produces predictions that are physically consistent with the underlying flow dynamics.

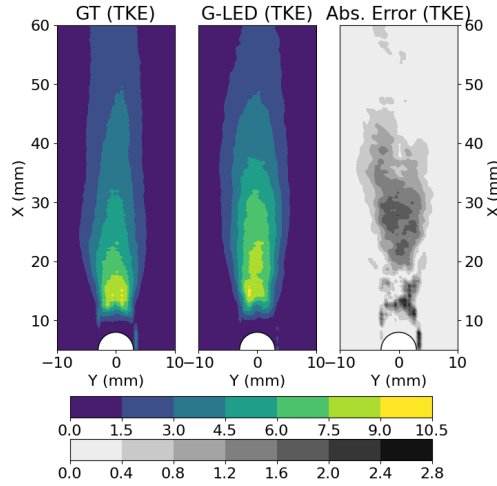


Figure 5.6: Comparison of TKE computed from the ground truth (left) and from the predicted dynamics (middle) from G-LED model. The right column shows the absolute error between prediction and ground truth.

5.3.2 Results from G-LED combined with POD

As illustrated in Fig. 5.2, G-LED can be integrated within the POD-DL framework. In contrast to the previous configuration, where training was performed directly on flow snapshots, this implementation utilizes POD coefficients as the training data, which are extracted by applying POD to the full dataset.

Similar to the previous case, POD coefficients derived from both steady-state regimes were utilized. To this end, POD was applied independently to the first 3200 snapshots from each regime, retaining the 128 most energetic modes. Consequently, the micro dynamics in this configuration are represented as vectors of dimension $\mathbf{v}_t \in \mathbb{R}^{1 \times 128}$.

Prior studies (Sengupta et al., 2025) have shown that only the coefficients corresponding to large-scale structures exhibit smooth, low-noise temporal behavior. Consistent with these findings and previous chapters, the standard POD-DL approach is effective at forecasting only a limited subset of these coefficients, making it unsuitable for predicting those associated with small-scale dynamics.

A key feature of G-LED lies in its ability to address this limitation: the encoder compresses the POD coefficient vector to a lower-dimensional macro representation, which is forecasted by the propagator, while the diffusion model generates the remaining coefficients to recover the full POD coefficient vector. In this implementation, the macro dynamics are encoded as $\mathbf{z}_t \in \mathbb{R}^{1 \times 32}$.

Figure 5.7 displays the evolution of the R^2 metric over the entire prediction horizon. Compared to the direct snapshot-based configuration (Fig. 5.3), the accuracy is reduced, reflecting the added challenge of reconstructing micro-scale information from compressed POD representations.

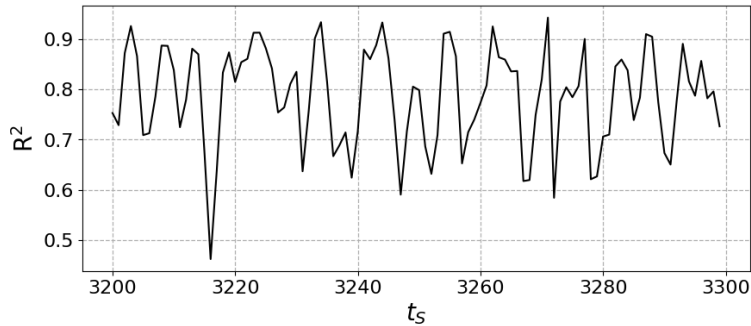


Figure 5.7: R^2 metric values, defined in eq. (5.12), for each prediction generated by G-LED when applied within the POD-DL methodology in the entire spatial domain.

This reduction in predictive accuracy is further illustrated in Fig. 5.8, where the most notable discrepancies appear in the mean vorticity. The same error is also clearly depicted in Fig. 5.9, which compares predicted and ground truth vorticity snapshots.

These visualizations reveal that although the retention of the 128 most energetic POD modes preserves structural information associated with small-scale features, the temporal dynamics of these modes are highly complex and irregular. As a result, their corresponding POD

coefficients are difficult to model accurately, limiting G-LED's ability to forecast the evolution of these fine-scale structures.

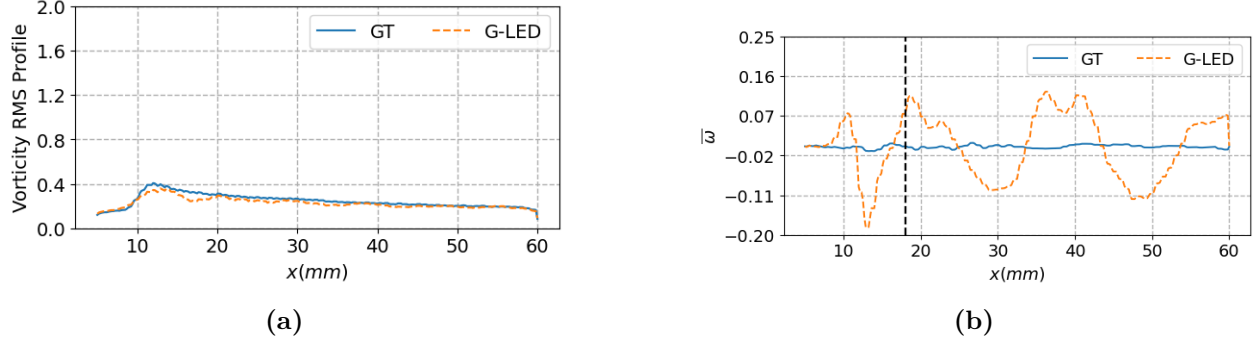


Figure 5.8: Comparison of the GT (solid line) and G-LED, within the POD-DL methodology, predictions (dashed lines) for the root mean square (RMS) vorticity profile along the streamwise direction (a) and the mean vorticity alongside the streamwise direction (b).

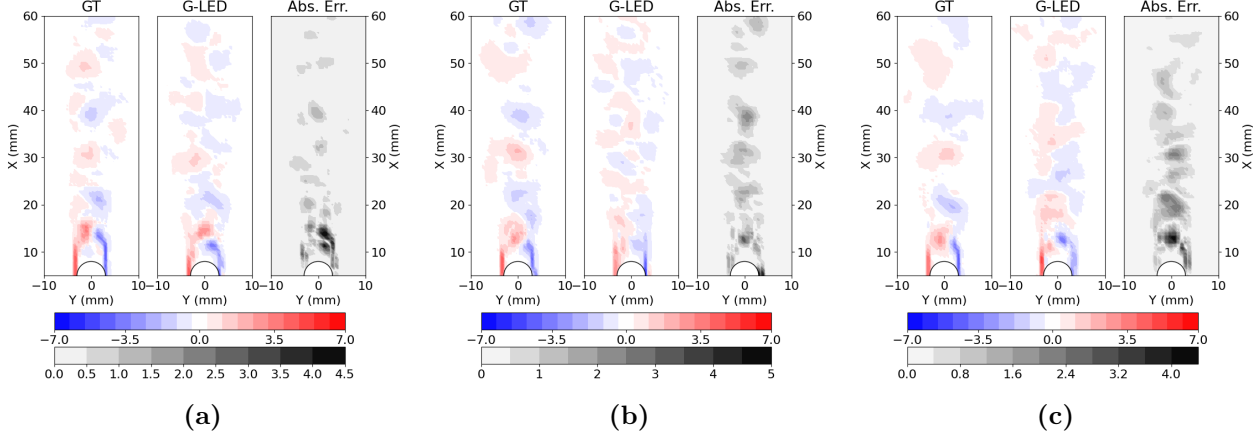


Figure 5.9: Comparison between the ground truth (GT) and predicted vorticity snapshots for snapshots (a) 3210, (b) 3250 and (c) 3300 corresponding to the second steady-state in the flow. In this case predictions were obtained from G-LED implemented in the POD-DL methodology.

Figure 5.10 presents a comparison of TKE, revealing that, in this case, G-LED predictions deviate from the expected physical behavior of the system. However, both this figure and Fig. 5.9 suggest that predictions associated with regions near the cylinder are more accurate than those corresponding to regions farther downstream.

This observation aligns with findings reported by Sengupta et al., 2025, who noted that the flow dynamics near the cylinder are predominantly captured by the most energetic POD modes. In contrast, the downstream flow, characterized by more chaotic, small-scale structures, is encoded in the lower-energy modes, which are inherently more difficult to model.

To further explore this effect, an additional case is examined in which POD is applied independently to two distinct spatial regions: one proximal and one distal to the cylinder, as schematically illustrated in Fig. 5.11.

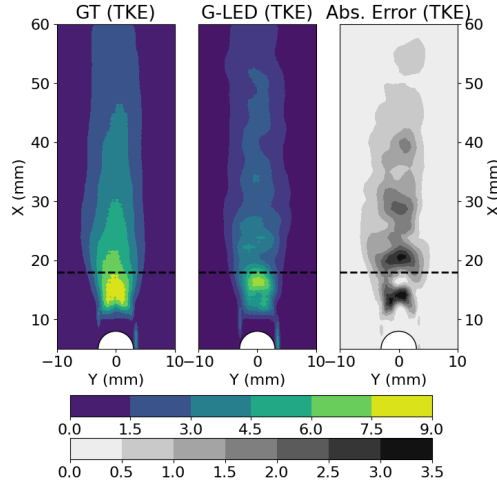


Figure 5.10: Comparison of TKE computed from the ground truth (left) and from the predicted dynamics (middle), corresponding to the application of the G-LED model within the POD-DL methodology. The right column shows the absolute error between prediction and ground truth.

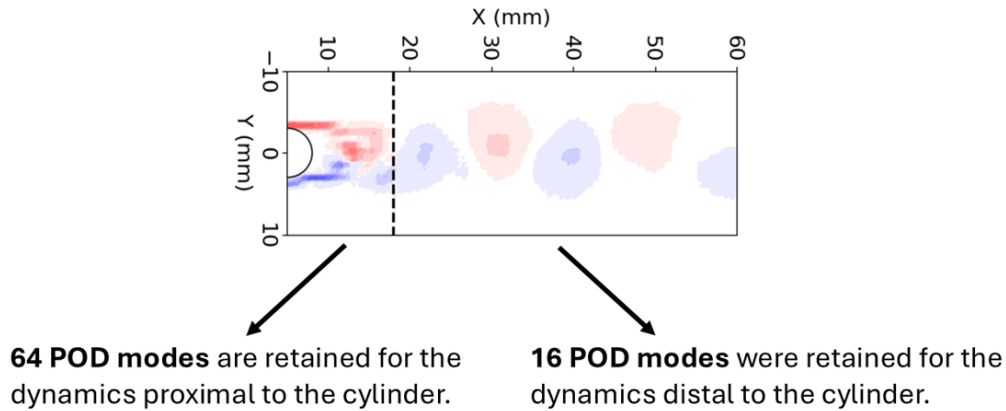


Figure 5.11: Illustration of the spatial domain partitioned into two disjoint subdomains, within which POD and G-LED methodologies are independently applied. Following the work of Sengupta et al., 2025 a larger number of modes is retained close the cylinder than in regions further downstream.

As illustrated in Fig. 5.11, the spatial domain is partitioned at the 17 mm mark, where POD is applied separately to each subdomain, retaining a different number of modes in each. In the region proximal to the cylinder, where the dynamics are less complex, a larger number of modes can be retained, 64 in this case, enabling the preservation of finer-scale features. Conversely, in the region farther from the cylinder, the flow becomes more intricate, necessitating a reduced set of modes for an accurate prediction; here, only 16 modes are retained. As a result, fine-scale structures can be predicted in the near-cylinder region, while only large-scale features are recoverable farther downstream.

Although fine-scale structures are not predicted in the entire domain, this experiment provides further insight into the effectiveness of G-LED in forecasting the temporal evolution of POD

coefficients. Consistent with previous cases, Fig. 5.12 reports the R^2 scores for all predictions.

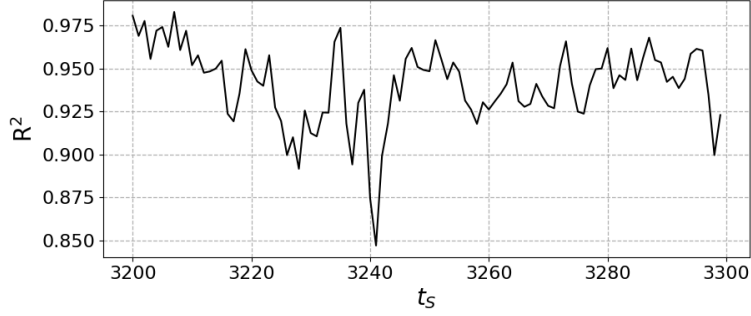


Figure 5.12: R^2 metric values, defined in eq. (5.12), for each prediction generated by G-LED when applied within the POD-DL methodology in two different areas of the spatial domain.

Figure 5.13 presents the RMS and mean vorticity profiles. A noticeable peak appears at the 17 mm mark, which coincides with the interface between the two spatial regions and reflects the error introduced by merging their respective POD representations. Despite this localized discrepancy, the predicted vorticity remains in close agreement with the ground truth across the rest of the domain. This behavior contrasts with the results shown in Fig. 5.8, where a more uniform deviation was observed throughout the domain, specially for the mean vorticity.

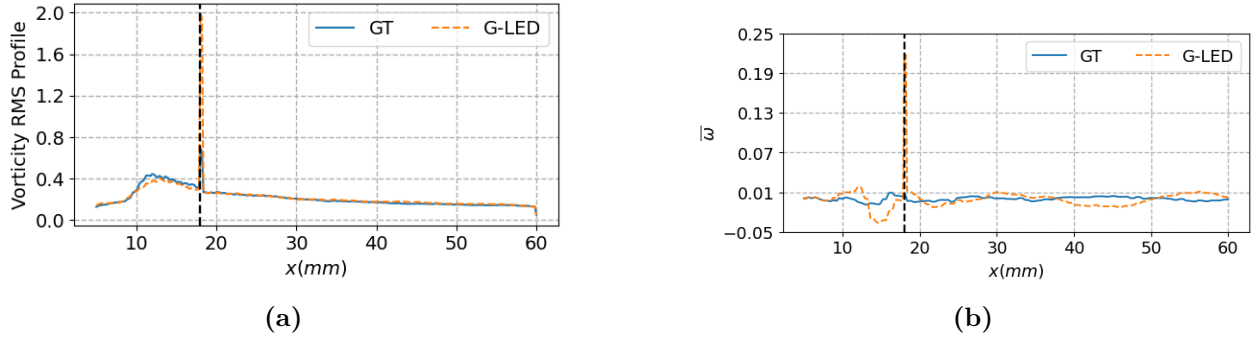


Figure 5.13: Comparison of the GT (solid line) and G-LED, within the POD-DL methodology, predictions (dashed lines) for the root mean square (RMS) vorticity profile along the streamwise direction (a) and the mean vorticity alongside the streamwise direction (b). In this case the spatial domain is divided into two disjoint subdomains, as depicted in Fig. 5.11, with the G-LED methodology applied independently to each.

The high predictive accuracy, excluding the localized discrepancy at the 17 mm interface, is visually confirmed in Fig. 5.14, which compares the predicted and GT vorticity fields. The model successfully reconstructs fine-scale structures in the near-cylinder region, while further downstream, where only large-scale modes are retained, the predictions remain temporally stable. Additionally, Fig. 5.15 compares the predicted and real TKE, denoted in eq. (4.5), demonstrating that in this cases G-LED yields physically consistent forecasts aligned with the dynamics of the underlying flow.

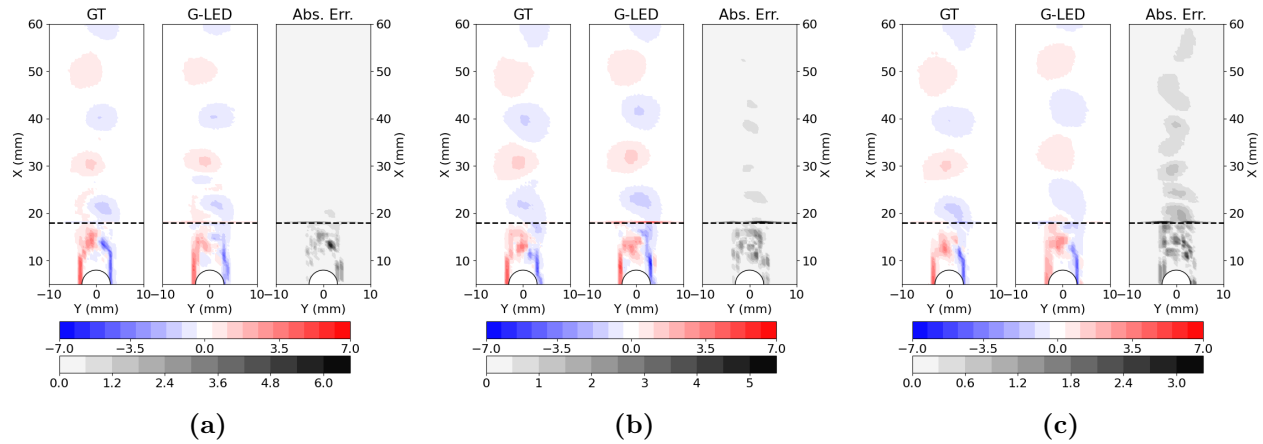


Figure 5.14: Comparison between the ground truth (GT) and predicted vorticity snapshots for snapshots (a) 3210, (b) 3250 and (c) 3300 corresponding to the second steady-state in the flow. In this case predictions were obtained from G-LED implemented in the POD-DL methodology and splitting the spatial domain in two disjoint areas as shown in Fig. 5.11.

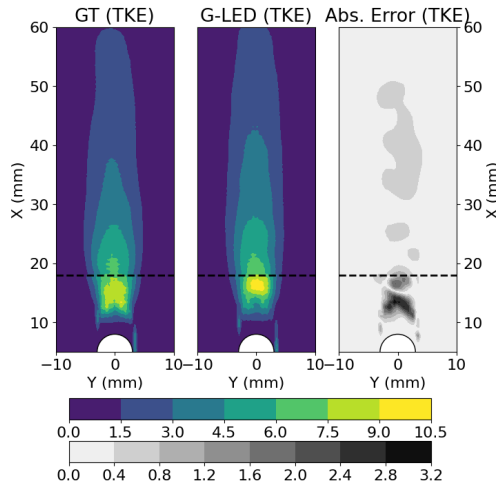


Figure 5.15: Comparison of TKE computed from the ground truth (left) and from the predicted dynamics (middle), corresponding to the application of the G-LED model within the POD-DL methodology. In this case the spatial domain is divided into two disjoint subdomains, as depicted in Fig. 5.11, with the G-LED methodology applied independently to each. The right column shows the absolute error between prediction and ground truth.

5.4 Discussion

The results presented in this chapter demonstrate that generative models such as G-LED hold considerable promise for forecasting small-scale structures in turbulent flows. This capability stems from their ability to approximate highly complex probability distributions, making them particularly well suited to applications where the goal is to reproduce the statistical properties of the flow, as is common in turbulence, rather than to perform one-to-one deterministic

predictions, as may be adequate in laminar regimes.

Generative models can be trained directly on high-dimensional snapshots of the flow, as is often done in climate forecasting applications (Kochkov et al., 2024), or alternatively on low-dimensional POD coefficients derived from such snapshots, as implemented in the POD-DL methodology.

In this chapter, it was shown that for the case of the three-dimensional turbulent wake of a cylinder, G-LED is capable of recovering small-scale structures when trained directly on flow snapshots. This can be qualitatively observed in Fig. 5.5. However, training such models is computationally intensive due to the high dimensionality of each training sample. Training required two RTX A4500 GPUs, one for the Transformer and one for the diffusion model, with the latter trained over approximately 17 hours and the former in 7 hours. In contrast, inference of 100 future flow fields required only about 3 minutes on a single RTX A4500 GPU, highlighting the model’s inference efficiency despite its costly training.

When G-LED is combined with POD, both the Transformer and the diffusion model are trained on POD coefficients, which are low-dimensional vectors. This greatly reduces the computational cost associated with high-dimensional training samples. Using the same two GPUs, the diffusion model was trained in approximately 8 hours and the Transformer in just 2 hours, achieving nearly two-times and four-times speed-up, respectively. In addition, the inference time for generating 100 future predictions was reduced to approximately 50 seconds.

However, this reduction in training cost comes at the expense of accuracy. As shown in Figs. 5.9 and 5.10, G-LED trained on POD coefficients fails to recover small-scale structures, particularly in regions far from the cylinder.

This difficulty is consistent with previous findings by Sengupta et al., 2025, who analyzed this same dataset. They reported that fine-scale structures are represented by low-energy POD modes, whose coefficients exhibit highly irregular, noise-like dynamics that are challenging to predict.

To address this limitation, an additional experiment was performed in which the spatial domain was partitioned into near- and far-cylinder regions, with POD applied separately to each individually. As shown in Fig. 5.11, a greater number of modes was retained in the near-cylinder region, where fine-scale structures are more predictable, while only large-scale features were retained farther downstream. This localized approach yielded stable temporal predictions, as reflected in Figs. 5.12 and 5.14.

The limited accuracy observed in G-LED predictions when trained on POD coefficients may therefore be attributed to the specific characteristics of this dataset. We do not rule out the potential for generative models to accurately predict low-energy POD coefficients in other scenarios, particularly in regions where their dynamics are less stochastic, as evidenced by the improved results near the cylinder.

Overall, G-LED has demonstrated a strong ability to reconstruct small-scale turbulence when trained directly on flow field snapshots. Future work will focus on developing more computationally efficient variants of this architecture to mitigate the high training costs currently required.

Chapter 6

Conclusions

This thesis has presented a comprehensive investigation into surrogate deep learning models for high-dimensional, spatio-temporal flow dynamics. The work has been organized into four key studies, each contributing distinct insights:

- The first chapter of this thesis introduced a hybrid ROM that combines POD with deep learning forecasting models, known as POD-DL (Abadía-Heredia et al., 2022). By projecting fluid flow data onto a lower-dimensional representation, formed by an orthonormal basis of POD modes and using a DNN to predict the temporal evolution of POD coefficients, this approach achieved efficient and accurate flow simulations. The hybrid POD-DL methodology was evaluated on canonical fluid dynamics problems, showing that it can capture dominant flow structures and forecast their evolution with significantly reduced computational cost compared to full-order CFD simulations.
- The second chapter conducted a systematic comparison between the POD-DL ROM and purely deep learning models, such as residual and variational autoencoders, for forecasting flow dynamics (Abadía-Heredia, Corrochano, et al., 2024). Using datasets representing a flow past a circular cylinder, at both laminar and turbulent regimes, to evaluate the models, it was found that while pure deep learning models can learn the dynamics in the training regime (especially for laminar flows), they struggle to generalize to more complex or unseen conditions. In contrast, the POD-DL ROM exhibited superior generalization capabilities and robustness, accurately predicting flow evolution across both laminar and turbulent conditions. The incorporation of modal decomposition provided the hybrid model with built-in physical insights and a compact state representation, enabling higher accuracy with less training data and fewer trainable parameters. This study highlighted the advantages of integrating physics-based features (POD modes) with deep learning, as the hybrid approach maintained predictive performance under conditions where purely data-driven models diverged or suffered accuracy loss.
- The third chapter proposed an adaptive forecasting pipeline to address long-term prediction in fluid flows (Abadía-Heredia et al., 2025). In this framework, the POD-DL model is used to autoregressively forecast future flow states, thus reducing the number

of expensive CFD computations, and is periodically retrained after a predefined number of predictions. We demonstrated that this adaptive POD-DL approach can significantly cut computational cost (achieving theoretical savings of 60%–80% in test cases) while maintaining accuracy. The model adapts to distribution shifts in the flow dynamics by incorporating a few newly observed snapshots and leveraging transfer learning to update the DNN. This on-the-fly retraining proved effective in halting forecast divergence when the flow underwent changes (e.g. a sudden alteration in boundary conditions or flow regime).

- The fourth and final chapter explored generative AI models for forecasting fine-scale turbulent structures in high-dimensional turbulent flows. This chapter is a continuation of the work developed by Gao, Kaltenbach, and Koumoutsakos, 2024 that proposed the Generative Learning of Effective Dynamics (G-LED) framework, which couples a low-dimensional latent evolution with a high-dimensional generative mapping, to fluid flow data. Here, instances of full-resolution turbulent flow fields were down-sampled or projected (either via learned autoencoders or via POD coefficients) into a lower-dimensional representation that evolves in time, and a generative decoder (based on diffusion models) was used to map the latent state back to the physical flow field. We compared training the generative model on raw snapshot data versus on POD coefficient data. The results showed training on raw snapshots allowed the model to learn directly from high-resolution data, preserving more fine-scale information, but at the expense of greater training complexity and data requirements. Using POD coefficients as training data, by contrast, simplified the learning task (since POD filters out minor scales and noise) and reduced the dimensionality of the output space, but it imposed a limit on the finest scales that could be reconstructed. This chapter highlighted the trade-off between fidelity and efficiency: the POD-DL generative method excelled at capturing large-scale coherent structures and delivered smooth long-term forecasts, whereas the full-field generative model (trained on raw data) better reproduced small-scale variability but was more computationally intensive.

6.1 Advances in modeling high-dimensional flows with POD-DL

Across this thesis, a unifying theme is the effectiveness of combining modal decomposition techniques with deep learning surrogate models to advance the modeling of high-dimensional, spatio-temporal flow fields. High-dimensional fluid dynamics systems, such as turbulent flows, pose a tremendous challenge for traditional modeling due to the curse of dimensionality and the wide range of spatial and temporal scales involved. The POD-DL methodology developed in this thesis addresses this challenge by leveraging the strengths of both data-driven learning, from deep learning models, and physics-based reduction, from modal decomposition.

By using POD to obtain a low-dimensional set of modes, we capture the dominant coherent structures within the flow. This not only reduces the problem size dramatically, by mapping high-dimensional tensors to low-dimensional vectors as shown in eq. (2.2), but also yields features that have physical interpretation, since each POD mode corresponds to a coherent flow

structure or pattern. Such physics-grounded features make the learning task more tractable and data-efficient, since the DNN only needs to learn the evolution of some significant modes rather than every fine detail of the flow, which is desirable in many practical applications (Jiménez, 2018). This hybrid POD-DL approach (Fig. 2.1) thus inherently filters out high-frequency noise and less-relevant features, focusing the model on the essential dynamics.

This thesis also showed that integrating POD with deep learning leads to improved generalization performance on complex flows compared to black-box models. For example, the hybrid POD-DL methodology outperformed purely deep learning architectures in predicting turbulent wake flows. The non-parametric nature of POD (computed via SVD as shown in eq. (2.10)) means it does not require training and can be computed even with limited data, providing a reliable basis that reduces the computational cost of training the DNN. Consequently, POD-DL model require fewer training samples to achieve a given accuracy, and it proved more robustness when extrapolating to flow conditions not seen in training. This is a significant advancement, where previous deep learning models often struggled with extrapolation in fluid dynamics, the hybrid approach maintained stability and accuracy by performing the predictions in the modal decomposition of the flow. These results underscore how incorporating prior physical knowledge (POD modes) into data-driven models can enhance reliability for high-dimensional physical systems.

A key advancement of the presented methodology is the potential to accelerate fluid simulations. By predicting multiple time-steps of the flow in the reduced-order space, instead of resolving them via CFD, the hybrid model functions as a fast surrogate model for the full solver. Chapter 4 presented an adaptive pipeline where the POD-DL model is interwoven with a traditional CFD solver, yielding substantial speed-ups (on the order of 60% – 80% less computation for long simulations) without significant loss of accuracy. This represents a practical step forward in model-order reduction: rather than a static ROM, we have a dynamic, adaptive ROM that can update itself and remain accurate over time.

On the other hand, traditional ROMs often struggle with fine-scale turbulence, as minor energetic modes are truncated. However, our exploration with generative models like G-LED, in Chapter 5, indicates that it is possible to recover the statistical effect of those fine scales through data-driven generation. This hybrid strategy, predicting the flow on a coarse manifold and using a generative decoder to inject small-scale details, is an innovative paradigm for high-dimensional modeling. It shows that deep learning models can be used not only as direct predictors but also as stochastic super-resolution tools to enhance coarse predictions with realistic fluctuations.

In summary, the combination of POD with deep learning as developed in this thesis contributes a powerful framework for modeling complex flow systems. It marries the interpretability and efficiency of classical ROMs with the expressiveness of state-of-the-art DNN. This synergy addresses some long-standing challenges in fluid dynamics modeling, achieving a level of performance and adaptability that neither approach could attain alone. The success of the POD-DL methodology in various scenarios (from laminar flows to fully resolved turbulence) demonstrates a versatile and scalable approach for high-dimensional spatio-temporal forecasting.

6.2 Limitations, challenges and future work

Despite the progress achieved, it is important to acknowledge the limitations and challenges that emerged during this research, identifying areas where further improvements are needed and guide the interpretation of the results.

One of the primary limitations encountered in this study was the scarcity of training data. While this issue was partially mitigated through the use of the POD-DL framework, which requires significantly fewer training samples, this reduction often comes at the cost of limiting predictions to the dominant flow structures, typically corresponding to the largest scales. This limitation arises because the POD coefficients associated with small-scale structures exhibit highly complex dynamics that are challenging to model.

Therefore exhibiting trade-offs in accuracy. By projecting onto a limited set of modes, the POD-DL model neglect higher-order dynamics that could cumulatively impact the flow evolution, especially over long times. In practice, we found that retaining a higher number of POD modes can improve accuracy but at the cost of a larger latent dimension, slowing down the DNN and increasing its training complexity, requiring more data to train. Similarly, the generative model experiments revealed a trade-off between capturing fine-scale dynamics and keeping the model tractable. The version trained on raw snapshots could reproduce small-scale turbulent fluctuations better than the POD version, but it was harder to train and more prone to slight instabilities, since modeling small details increases the risk of error accumulation. On the other hand, the POD hybrid model produced smoother predictions but sometimes under-predicted the intensity of fine-scale turbulence, as expected when minor energy modes are truncated. These accuracy trade-offs highlight that there is no one-size-fits-all solution, the ideal model complexity depends on the required fidelity and the specific application. For instance, if statistical accuracy is the goal (e.g. correct energy spectrum or mean flow), a coarser model like the POD-DL may suffice, but if instantaneous detailed flow structure is needed, a more complex or high-dimensional model like G-LED may be necessary. Balancing these needs is a persistent challenge.

Another challenge related to deep learning models, is that they are not immune to distribution shifts and extrapolation errors. For instance, within the POD-DL framework, a fixed POD basis might fail to represent new flow states that were not present in the training data (e.g. a new vortex structure appears, the flow enters a transitory state, a change of attractor, etc.).

Through this thesis, it was observed that when the flow dynamics diverged from the training manifold, the POD-DL model's predictions eventually deteriorate. Although, this was partially addressed by introducing an adaptive retraining, this assumes intermittent access to ground truth data, and for highly complex scenario like the experimental turbulent flow the prediction window could be very short due to the frequent changes in the dynamics distribution. Thus, ensuring reliability over long-term predictions remains an open research line.

Following the line of the adaptive model, this thesis did not explore uncertainty quantification (UQ) techniques to assess the reliability of predictions. Yet, such capabilities are essential for enabling an adaptive forecasting framework that can autonomously adjust the prediction horizon based on deviations between the predicted and true flow dynamics. Several approaches

have been proposed in the literature to address this need, including conformal prediction and Bayesian neural networks. For instance, recent studies have employed conformal prediction in conjunction with modern Hopfield networks to construct calibrated prediction intervals for time-series forecasts. Integrating uncertainty quantification methods into the present modeling pipeline would enable the system not only to forecast the evolution of turbulent flows but also to indicate when predictions may become unreliable. As such functionality was not implemented in this work, it represents a promising direction for future research.

Although the thesis demonstrated significant improvements in modeling high-dimensional flows, these limitations mitigate the results. They suggest that while hybrid and deep learning models are powerful, careful consideration must be given to their training regimes, validation and integration with domain knowledge to ensure robust and trustworthy deployment.

Finally, note that the intersection of deep learning and fluid dynamics is a rapidly evolving field. As computational power grows and algorithms improve, the approaches demonstrated in this thesis can be extended to increasingly complex and realistic flows. By continuously incorporating new developments in machine learning (e.g. novel network architectures, better training strategies) and staying grounded in fluid physics, we anticipate that data-driven flow modeling will become an indispensable tool alongside traditional analysis and simulation. The hybrid models presented here serve as a foundation, and the future work outlined above will drive them closer to the ultimate goal: fast, accurate, and reliable modeling of high-dimensional flow dynamics for both understanding fundamental physics and enabling advanced engineering solutions.

References

- Abadía-Heredia, R., López-Martín, M., Carro, B., Arribas, J., Pérez, J., & Le Clainche, S. (2022). A predictive hybrid reduced order model based on proper orthogonal decomposition combined with deep learning architectures. *Expert Systems with Applications*, 187, 115910. <https://doi.org/https://doi.org/10.1016/j.eswa.2021.115910>
- Abadía-Heredia, R., Corrochano, A., Lopez-Martin, M., & Le Clainche, S. (2024). Generalization capabilities and robustness of hybrid machine learning models grounded in flow physics compared to purely deep learning models.
- Abadía-Heredia, R., Lopez-Martin, M., & Clainche, S. L. (2025). An adaptive framework for autoregressive forecasting in cfd using hybrid modal decomposition and deep learning.
- Abadía-Heredia, R., Lopez-Martín, M., & le Clainche, S. (2024). Evaluating forecast divergence in fluid dynamics: A metric for autoregressive models in the absence of true solution data. *ECCOMAS 2024*.
- Abadía-Heredia, R., Pariente, A., Pérez, J. M., & Le Clainche, S. (2021). Tortuosity in tumours: The need of combining multi-phase flows with machine learning tools. *Results in Engineering*, 11, 100234. <https://doi.org/https://doi.org/10.1016/j.rineng.2021.100234>
- Alfonsi, G. (2009). Reynolds-averaged navier–stokes equations for turbulence modeling. *Applied Mechanics Reviews*, 62(4), 040802. <https://doi.org/10.1115/1.3124648>
- Ali, N., Viggiano, B., Tutkun, M., & Cal, R. B. (2021). Forecasting the evolution of chaotic dynamics of two-phase slug flow regime. *Journal of Petroleum Science and Engineering*, 205, 108904. <https://doi.org/https://doi.org/10.1016/j.petrol.2021.108904>
- Anthropic. (2024). The claude 3 model family: Opus, sonnet, haiku.
- Antoulas, A. C., Sorensen, D. C., & Gugercin, S. (2001). A survey of model reduction methods for large-scale systems. In *Structured matrices in mathematics, computer science, and engineering, I (Boulder, CO, 1999)* (pp. 193–219, Vol. 280). Amer. Math. Soc., Providence, RI. <https://doi.org/10.1090/conm/280/04630>
- Aubry, N., Guyonnet, R., & Lima, R. (1991). Spatiotemporal analysis of complex signals: Theory and applications. *J Stat Phys*, 64, 683–739. <https://doi.org/10.1007/BF01048312>
- Auer, A., Gauch, M., Klotz, D., & Hochreiter, S. (2023). Conformal prediction for time series with modern hopfield networks.
- Bae, H. J., & Koumoutsakos, P. (2022). Scientific multi-agent reinforcement learning for wall-models of turbulent flows. *Nature Communications*, 13(1). <https://doi.org/10.1038/s41467-022-28957-7>
- Baldi, P. (2021). *Deep learning in science*. Cambridge University Press.

- Barckley, D., & Henderson, R. (1996). Three-dimensional floquet stability analysis of the wake of a circular cylinder. *J. Fluid Mech.*, 322, 215–241.
- Beck, M., Pöppel, K., Spanring, M., Auer, A., Prudnikova, O., Kopp, M., Klambauer, G., Brandstetter, J., & Hochreiter, S. (2024). Xlstm: Extended long short-term memory.
- Begiasvili, B., Groun, N., Garicano-Mena, J., Le Clainche, S., & Valero, E. (2023). Data-driven modal decomposition methods as feature detection techniques for flow problems: A critical assessment. *Physics of Fluids*, 35(4), 041301. <https://doi.org/10.1063/5.0142102>
- Beltrán, V., Clainche, S. L., & Vega, J. M. (2020). A data-driven rom based on hodmd. In F. Martínez Álvarez, A. Troncoso Lora, J. A. Sáez Muñoz, H. Quintián, & E. Corchado (Eds.), *14th international conference on soft computing models in industrial and environmental applications (soco 2019)* (pp. 567–576). Springer International Publishing.
- Beltrán, V., Le Clainche, S., & Vega, J. M. (2022). An adaptive data-driven reduced order model based on higher order dynamic mode decomposition. *Journal of Scientific Computing*, 92(1). <https://doi.org/10.1007/s10915-022-01855-2>
- Benner, P., Gugercin, S., & Willcox, K. (2015). A survey of projection-based model reduction methods for parametric dynamical systems. *SIAM Review*, 57(4), 483–531. <https://doi.org/10.1137/130932715>
- Berkooz, G., Holmes, P., & Lumley, J. L. (1993). The proper orthogonal decomposition in the analysis of turbulent flows. *Annual Review of Fluid Mechanics*, 25(Volume 25, 1993), 539–575. <https://doi.org/10.1146/annurev.fl.25.010193.002543>
- Besabe, L., Girfoglio, M., Quaini, A., & Rozza, G. (2025). Data-driven reduced order modeling of a two-layer quasi-geostrophic ocean model. *Results in Engineering*, 25, 103691. <https://doi.org/10.1016/j.rineng.2024.103691>
- Blackburn, H. M., & Henderson, R. (1999). A study of two-dimensional flow past an oscillating cylinder. *J. Fluid Mech.*, 385, 255–286.
- Blackburn, H., Lee, D., Albrecht, T., & Singh, J. (2019). Semtex: A spectral element-fourier solver for the incompressible navier-stokes equations in cylindrical or cartesian coordinates. *Computer Physics Communications*, 245, 106804. <https://doi.org/10.1016/j.cpc.2019.05.015>
- Blei, D. M., Kucukelbir, A., & McAuliffe, J. D. (2017). Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518), 859–877. <https://doi.org/10.1080/01621459.2017.1285773>
- Bourgeois, J. A., Noack, B. R., & Martinuzzi, R. J. (2013). Generalized phase average with applications to sensor-based flow estimation of the wall-mounted square cylinder wake. *Journal of Fluid Mechanics*, 736, 316–350. <https://doi.org/10.1017/jfm.2013.494>
- Brès, G. A., Jordan, P., Jaunet, V., Le Rallic, M., Cavalieri, A. V. G., Towne, A., Lele, S. K., Colonius, T., & Schmidt, O. T. (2018). Importance of the nozzle-exit boundary-layer state in subsonic turbulent jets. *Journal of Fluid Mechanics*, 851, 83–124. <https://doi.org/10.1017/jfm.2018.476>
- Briden, J., Johnson, B. J., Linares, R., & Cauligi, A. (2025). Diffusion policies for generative modeling of spacecraft trajectories. In *Aiaa scitech 2025 forum*. <https://doi.org/10.2514/6.2025-2775>
- Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G.,

- Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., ... Amodei, D. (2020). Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, & H. Lin (Eds.), *Advances in neural information processing systems* (pp. 1877–1901, Vol. 33). Curran Associates, Inc.
- Brunton, S. L., Noack, B. R., & Koumoutsakos, P. (2020). Machine learning for fluid mechanics. *Annual Review of Fluid Mechanics*, 52(1), 477–508. <https://doi.org/10.1146/annurev-fluid-010719-060214>
- Buaria, D., & Sreenivasan, K. R. (2023). Forecasting small-scale dynamics of fluid turbulence using deep neural networks. *Proceedings of the National Academy of Sciences*, 120(30), e2305765120. <https://doi.org/10.1073/pnas.2305765120>
- Carrasco, A., Rodriguez-Fernandez, V., & Linares, R. (2025). Large language models as autonomous spacecraft operators in kernal space program. *Advances in Space Research*. <https://doi.org/https://doi.org/10.1016/j.asr.2025.06.034>
- Chan, S. H. (2025). Tutorial on diffusion models for imaging and vision.
- Cheng, M., Fang, F., Pain, C., & Navon, I. (2020). Data-driven modelling of nonlinear spatio-temporal fluid flows using a deep convolutional generative adversarial network. *Computer Methods in Applied Mechanics and Engineering*, 365, 113000. <https://doi.org/https://doi.org/10.1016/j.cma.2020.113000>
- Chorin, A. J., & Marsden, J. E. (1979). *A mathematical introduction to fluid mechanics*. Springer US. <https://doi.org/10.1007/978-1-4684-0082-3>
- Corrochano, A., Xavier, D., Schlatter, P., Vinuesa, R., & Le Clainche, S. (2021). Flow structures on a planar food and drug administration (fda) nozzle at low and intermediate reynolds number. *Fluids*, 6(1), 4.
- Corrochano, A., Freitas, R. S. M., Parente, A., & Clainche, S. L. (2023). A predictive physics-aware hybrid reduced order model for reacting flows.
- De Bortoli, V., Thornton, J., Heng, J., & Doucet, A. (2021). Diffusion schrödinger bridge with applications to score-based generative modeling. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, & J. W. Vaughan (Eds.), *Advances in neural information processing systems* (pp. 17695–17709, Vol. 34). Curran Associates, Inc.
- DeepSeek-AI, Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., Zhu, Q., Ma, S., Wang, P., Bi, X., Zhang, X., Yu, X., Wu, Y., Wu, Z. F., Gou, Z., Shao, Z., Li, Z., Gao, Z., ... Zhang, Z. (2025). Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning.
- DeMont, E., & Gosline, J. (1988). Mechanics of jet propulsion in the hydromedusan jellyfish, *polyorchis penicillatus*. *J. Experimental Biology*, 134, 347–361.
- Dhariwal, P., & Nichol, A. (2021). Diffusion models beat gans on image synthesis. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, & J. W. Vaughan (Eds.), *Advances in neural information processing systems* (pp. 8780–8794, Vol. 34). Curran Associates, Inc.
- Doran, P. M. (2013). Chapter 7 - fluid flow. In P. M. Doran (Ed.), *Bioprocess engineering principles (second edition)* (Second Edition, pp. 201–254). Academic Press. <https://doi.org/https://doi.org/10.1016/B978-0-12-220851-5.00007-1>
- Du, P., Parikh, M. H., Fan, X., Liu, X.-Y., & Wang, J.-X. (2024). Conditional neural field latent diffusion model for generating spatiotemporal turbulence. *Nature Communications*, 15(1), 10416. <https://doi.org/10.1038/s41467-024-54712-1>

- Ehlert, A., Nayeri, C. N., Morzynski, M., & Noack, B. R. (2019). Locally linear embedding for transient cylinder wakes.
- Eivazi, H., Le Clainche, S., Hoyas, S., & Vinuesa, R. (2022). Towards extraction of orthogonal and parsimonious non-linear modes from turbulent flows. *Expert Systems with Applications*, 202, 117038. <https://doi.org/https://doi.org/10.1016/j.eswa.2022.117038>
- Ekambaram, V., Jati, A., Nguyen, N., Sinthong, P., & Kalagnanam, J. (2023). Tsmixer: Lightweight mlp-mixer model for multivariate time series forecasting. *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 459–469. <https://doi.org/10.1145/3580305.3599533>
- Ferrari, M., Werner, G., Bahrmann, P., Richartz, B., & Figulla, H. (2006). Turbulent flow as a cause for underestimating coronary flow reserve measured by doppler guide wire. *Cardiovascular Ultrasound*, 14(4).
- Foggi Rota, G., Amor, C., Le Clainche, S., & Rosti, M. E. (2024). Unified view of elastic and elasto-inertial turbulence in channel flows at low and moderate reynolds numbers. *Phys. Rev. Fluids*, 9, L122602. <https://doi.org/10.1103/PhysRevFluids.9.L122602>
- Fotiadis, S., Brenowitz, N., Geffner, T., Cohen, Y., Pritchard, M., Vahdat, A., & Mardani, M. (2024). Stochastic flow matching for resolving small-scale physics.
- Fukami, K., Fukagata, K., & Taira, K. (2020). Assessment of supervised machine learning methods for fluid flows. *Theoretical and Computational Fluid Dynamics*, 34(4), 497–519. <https://doi.org/10.1007/s00162-020-00518-y>
- Fukami, K., Nakamura, T., & Fukagata, K. (2020). Convolutional neural network based hierarchical autoencoder for nonlinear mode decomposition of fluid field data. *Physics of Fluids*, 32(9), 095110. <https://doi.org/10.1063/5.0020721>
- Gao, H., Han, X., Fan, X., Sun, L., Liu, L.-P., Duan, L., & Wang, J.-X. (2024). Bayesian conditional diffusion models for versatile spatiotemporal turbulence generation. *Computer Methods in Applied Mechanics and Engineering*, 427, 117023. <https://doi.org/https://doi.org/10.1016/j.cma.2024.117023>
- Gao, H., Kaltenbach, S., & Koumoutsakos, P. (2024). Generative learning for forecasting the dynamics of high-dimensional complex systems. *Nature Communications*, 15(1), 8904. <https://doi.org/10.1038/s41467-024-53165-w>
- Geneva, N., & Zabararas, N. (2022). Transformers for modeling physical systems. *Neural Networks*, 146, 272–289. <https://doi.org/https://doi.org/10.1016/j.neunet.2021.11.022>
- Giral, F., Gómez, I., Vinuesa, R., & Clainche, S. L. (2025). Transformer-based fault-tolerant control for fixed-wing uavs using knowledge distillation and in-context adaptation.
- Glorot, X., & Bengio, Y. (2010, 13–15 May). Understanding the difficulty of training deep feedforward neural networks. In Y. W. Teh & M. Titterton (Eds.), *Proceedings of the thirteenth international conference on artificial intelligence and statistics* (pp. 249–256, Vol. 9). PMLR.
- Gneiting, T., & Katzfuss, M. (2014). Probabilistic forecasting. *Annual Review of Statistics and Its Application*, 1(1), 125–151. <https://doi.org/10.1146/annurev-statistics-062713-085831>
- Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning* [<http://www.deeplearningbook.org>]. MIT Press.
- Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative adversarial nets. In Z. Ghahramani, M. Welling,

- C. Cortes, N. Lawrence, & K. Weinberger (Eds.), *Advances in neural information processing systems* (Vol. 27). Curran Associates, Inc.
- Han, R., Wang, Y., Zhang, Y., & Chen, G. (2019). A novel spatial-temporal prediction method for unsteady wake flows based on hybrid deep neural network. *Physics of Fluids*, 31(12), 127101. <https://doi.org/10.1063/1.5127247>
- Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., ... Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, 585(7825), 357–362. <https://doi.org/10.1038/s41586-020-2649-2>
- Hasegawa, K., Fukami, K., Murata, T., & Fukagata, K. (2020). Machine-learning-based reduced-order modeling for unsteady flows around bluff bodies of various shapes. *Theoretical and Computational Fluid Dynamics*, 34(4), 367–383. <https://doi.org/10.1007/s00162-020-00528-w>
- He, K., Zhang, X., Ren, S., & Sun, J. (2014). Spatial pyramid pooling in deep convolutional networks for visual recognition. In D. Fleet, T. Pajdla, B. Schiele, & T. Tuytelaars (Eds.), *Computer vision – eccv 2014* (pp. 346–361). Springer International Publishing. https://doi.org/https://doi.org/10.1007/978-3-319-10578-9_23
- He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *2015 IEEE International Conference on Computer Vision (ICCV)*, 1026–1034. <https://doi.org/10.1109/ICCV.2015.123>
- He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778. <https://doi.org/10.1109/CVPR.2016.90>
- Heimbach, L., Kaltenbach, S., Karnakov, P., Alexander, F. J., & Koumoutsakos, P. (2025). Reinforcement learning closures for underresolved partial differential equations using synthetic data.
- Ho, J., Chan, W., Saharia, C., Whang, J., Gao, R., Gritsenko, A., Kingma, D. P., Poole, B., Norouzi, M., Fleet, D. J., & Salimans, T. (2022). Imagen video: High definition video generation with diffusion models.
- Ho, J., Jain, A., & Abbeel, P. (2020). Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, & H. Lin (Eds.), *Advances in neural information processing systems* (pp. 6840–6851, Vol. 33). Curran Associates, Inc.
- Ho, J., Salimans, T., Gritsenko, A., Chan, W., Norouzi, M., & Fleet, D. J. (2022). Video diffusion models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, & A. Oh (Eds.), *Advances in neural information processing systems* (pp. 8633–8646, Vol. 35). Curran Associates, Inc.
- Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, 9(8), 1735–1780. <https://doi.org/10.1162/neco.1997.9.8.1735>
- Holmes, P., Lumley, J. L., & Berkooz, G. (1996). *Turbulence, coherent structures, dynamical systems and symmetry*. Cambridge University Press.
- Jackson, C. P. (1987). A finite-element study of the onset of vortex shedding in flow past variously shaped bodies. *J. Fluid Mech.*, 182, 23–45.
- Jacobsen, C., Zhuang, Y., & Duraisamy, K. (2024). Cocogen: Physically-consistent and conditioned score-based generative models for forward and inverse problems.

- Januschowski, T., Gasthaus, J., Wang, Y., Salinas, D., Flunkert, V., Bohlke-Schneider, M., & Callot, L. (2020). Criteria for classifying forecasting methods [M4 Competition]. *International Journal of Forecasting*, 36(1), 167–177. <https://doi.org/10.1016/j.ijforecast.2019.05.008>
- Jiménez, J. (2018). Coherent structures in wall-bounded turbulence. *Journal of Fluid Mechanics*, 842, P1. <https://doi.org/10.1017/jfm.2018.144>
- Kani, J. N., & Elsheikh, A. H. (2017). Dr-rnn: A deep residual recurrent neural network for model reduction.
- Khurana, D., Koli, A., Khatter, K., & Singh, S. (2023). Natural language processing: State of the art, current trends and challenges. *Multimedia Tools and Applications*, 82(3), 3713–3744. <https://doi.org/10.1007/s11042-022-13428-4>
- Kičić, I., Vlachas, P. R., Arampatzis, G., Chatzimanolakis, M., Guibas, L., & Koumoutsakos, P. (2023). Adaptive learning of effective dynamics for online modeling of complex systems. *Computer Methods in Applied Mechanics and Engineering*, 415, 116204. <https://doi.org/10.1016/j.cma.2023.116204>
- Kingma, D., Salimans, T., Poole, B., & Ho, J. (2021). Variational diffusion models. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, & J. W. Vaughan (Eds.), *Advances in neural information processing systems* (pp. 21696–21707, Vol. 34). Curran Associates, Inc.
- Kingma, D. P., & Ba, J. (2017). Adam: A method for stochastic optimization.
- Kingma, D. P., & Welling, M. (2013). Auto-encoding variational bayes.
- Kochkov, D., Yuval, J., Langmore, I., Norgaard, P., Smith, J., Mooers, G., Klöwer, M., Lottes, J., Rasp, S., Düben, P., Hatfield, S., Battaglia, P., Sanchez-Gonzalez, A., Willson, M., Brenner, M. P., & Hoyer, S. (2024). Neural general circulation models for weather and climate. *Nature*, 632(8027), 1060–1066. <https://doi.org/10.1038/s41586-024-07744-y>
- Kohl, G., Chen, L.-W., & Thuerey, N. (2024). Benchmarking autoregressive conditional diffusion models for turbulent flow simulation.
- Kurz, M., Kempf, D., Blind, M. P., Kopper, P., Offenhäuser, P., Schwarz, A., Starr, S., Keim, J., & Beck, A. (2025). Galæxi: Solving complex compressible flows with high-order discontinuous galerkin methods on accelerator-based systems. *Computer Physics Communications*, 306, 109388. <https://doi.org/10.1016/j.cpc.2024.109388>
- Kutz, J. N. (2017). Deep learning in fluid dynamics. *Journal of Fluid Mechanics*, 814, 1–4. <https://doi.org/10.1017/jfm.2016.803>
- Lakshminarayanan, B., Pritzel, A., & Blundell, C. (2017). Simple and scalable predictive uncertainty estimation using deep ensembles.
- Lazpita, E., Mares, A., Quintero, P., Garicano-Mena, J., & Le Clainche, S. (2024). Modeling heart flow dynamics using numerical simulations to identify the vortex ring: A practical guide. *Results in Engineering*, 24, 103644. <https://doi.org/10.1016/j.rineng.2024.103644>
- Le Clainche, S., Pérez, J. M., & Vega, J. M. (2018). Spatio-temporal flow structures in the three-dimensional wake of a circular cylinder. *Fluid Dyn. Res.*, 50(5), 051406.
- Le Clainche, S., Benocci, C., & Parente, A. (2014a). Principal component analysis on a les of a squared ribbed channel. In Á. Herrero, B. Baroque, F. Klett, A. Abraham, V. Snášel, A. C. de Carvalho, P. G. Bringas, I. Zelinka, H. Quintián, & E. Corchado

- (Eds.), *International joint conference soco'13-cisis'13-iceute'13* (pp. 259–268). Springer International Publishing. https://doi.org/doi.org/10.1007/978-3-319-01854-6_27
- Le Clainche, S., Benocci, C., & Parente, A. (2014b). Principal component analysis on a les of a squared ribbed channel. In Á. Herrero, B. Baruque, F. Klett, A. Abraham, V. Snášel, A. C. de Carvalho, P. G. Bringas, I. Zelinka, H. Quintián, & E. Corchado (Eds.), *International joint conference soco'13-cisis'13-iceute'13* (pp. 259–268). Springer International Publishing.
- Le Clainche, S., & Vega, J. M. (2017). Higher order dynamic mode decomposition to identify and extrapolate flow patterns. *Physics of Fluids*, 29(8), 084102. <https://doi.org/10.1063/1.4997206>
- Lin, L., Li, Z., Li, R., Li, X., & Gao, J. (2023). Diffusion models for time series applications: A survey.
- Lin, L., Li, Z., Li, R., Li, X., & Gao, J. (2024a). Diffusion models for time-series applications: A survey. *Frontiers of Information Technology & Electronic Engineering*, 25(1), 19–41. <https://doi.org/10.1631/FITEE.2300310>
- Lin, L., Li, Z., Li, R., Li, X., & Gao, J. (2024b). Diffusion models for time-series applications: A survey. *Frontiers of Information Technology & Electronic Engineering*, 25(1), 19–41. <https://doi.org/10.1631/FITEE.2300310>
- Lippe, P., Veeling, B., Perdikaris, P., Turner, R., & Brandstetter, J. (2023a). Pde-refiner: Achieving accurate long rollouts with neural pde solvers. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, & S. Levine (Eds.), *Advances in neural information processing systems* (pp. 67398–67433, Vol. 36). Curran Associates, Inc.
- Lippe, P., Veeling, B., Perdikaris, P., Turner, R., & Brandstetter, J. (2023b). Pde-refiner: Achieving accurate long rollouts with neural pde solvers. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, & S. Levine (Eds.), *Advances in neural information processing systems* (pp. 67398–67433, Vol. 36). Curran Associates, Inc.
- Liu, J. T. C. (1988). Contributions to the understanding of large-scale coherent structures in developing free turbulent shear flows. In J. W. Hutchinson & T. Y. Wu (Eds.), *Advances in applied mechanics* (pp. 183–309, Vol. 26). Elsevier. [https://doi.org/https://doi.org/10.1016/S0065-2156\(08\)70289-8](https://doi.org/https://doi.org/10.1016/S0065-2156(08)70289-8)
- Loshchilov, I., & Hutter, F. (2017). Sgdr: Stochastic gradient descent with warm restarts.
- Luchtenburg, D., Noack, B., & Schlegel, M. (2009a). Galerkin method for fluid flows with analytical examples and matlab source codes. *Berlin Institute of Technology. Report 01*.
- Luchtenburg, D., Noack, B., & Schlegel, M. (2009b). An introduction to the pod galerkin method for fluid flows with analytical examples and matlab source codes. *Berlin Institute of Technology MB1, Muller-Breslau-Strabe, 11*.
- Lumley, J. L. (1967). The structure of inhomogeneous turbulent flows. In A. M. Yaglom & V. I. Tartarsky (Eds.), *Atmospheric turbulence and radio wave propagation* (pp. 166–178). Nauka.
- Lundberg, S. M., & Lee, S.-I. (2017). A unified approach to interpreting model predictions. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, & R. Garnett (Eds.), *Advances in neural information processing systems* (Vol. 30). Curran Associates, Inc.

- Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2018). The m4 competition: Results, findings, conclusion and way forward. *International Journal of Forecasting*, 34(4), 802–808. <https://doi.org/https://doi.org/10.1016/j.ijforecast.2018.06.001>
- Martínez-Sánchez, A., López, E., Le Clainche, S., Lozano-Durán, A., Srivastava, A., & Vinuesa, R. (2023). Causality analysis of large-scale structures in the flow around a wall-mounted square cylinder. *Journal of Fluid Mechanics*, 967, A1. <https://doi.org/10.1017/jfm.2023.423>
- Martínez-Sánchez, Á., Lazpita, E., Corrochano, A., Le Clainche, S., Hoyas, S., & Vinuesa, R. (2023). Data-driven assessment of arch vortices in simplified urban flows. *International Journal of Heat and Fluid Flow*, 100, 109101. <https://doi.org/https://doi.org/10.1016/j.ijheatfluidflow.2022.109101>
- Marusic, I., Candler, G., Interrante, V., Subbareddy, P., & Moss, A. (2003). Real time feature extraction for the analysis of turbulent flows. *Semantic Scholar*.
- Mata, L., Abadía-Heredia, R., Lopez-Martin, M., Pérez, J. M., & Le Clainche, S. (2023). Forecasting through deep learning and modal decomposition in two-phase concentric jets. *Expert Systems with Applications*, 232, 120817. <https://doi.org/https://doi.org/10.1016/j.eswa.2023.120817>
- Maulik, R., Fukami, K., Ramachandra, N., Fukagata, K., & Taira, K. (2020). Probabilistic neural networks for fluid flow surrogate modeling and data recovery. *Phys. Rev. Fluids*, 5, 104401. <https://doi.org/10.1103/PhysRevFluids.5.104401>
- Mendez, M. A., Balabane, M., & Buchlin, J.-M. (2019). Multi-scale proper orthogonal decomposition of complex fluid flows. *Journal of Fluid Mechanics*, 870, 988–1036. <https://doi.org/10.1017/jfm.2019.212>
- Mendez, M. A., Hess, D., Watz, B. B., & Buchlin, J.-M. (2020). Multiscale proper orthogonal decomposition (mpod) of tr-piv data—a case study on stationary and transient cylinder wake flows. *Measurement Science and Technology*, 31(9), 094014. <https://doi.org/10.1088/1361-6501/ab82be>
- Milano, M., & Koumoutsakos, P. (2002). Neural network modeling for near wall turbulent flow. *Journal of Computational Physics*, 182(1), 1–26. <https://doi.org/https://doi.org/10.1006/jcph.2002.7146>
- Mohan, A. T., & Gaitonde, D. V. (2018). A deep learning based approach to reduced order modeling for turbulent flow control using lstm neural networks.
- Moin, P. (2002). Advances in large eddy simulation methodology for complex flows. *International Journal of Heat and Fluid Flow*, 23(5), 710–720. [https://doi.org/https://doi.org/10.1016/S0142-727X\(02\)00167-4](https://doi.org/https://doi.org/10.1016/S0142-727X(02)00167-4)
- Muijres, F. T., Lentink, D., Oudheusden, B. v., Donker-Duyvis, F. J., & Leeuwen, J. v. (2005). Visualization of vortex wake interactions of a flapping foil. *Proceedings of PSFVIP-5, 5th Pacific Symposium on Flow Visualization and Image Processing, Daydream Island, Australia, 27-29 September 2005*, 7.1.1–7.1.13.
- Muñoz, E., Dave, H., D'Alessio, G., Bontempi, G., Parente, A., & Le Clainche, S. (2023). Extraction and analysis of flow features in planar synthetic jets using different machine learning techniques. *Physics of Fluids*, 35(9), 094107. <https://doi.org/10.1063/5.0163833>

- Murata, T., Fukami, K., & Fukagata, K. (2020). Nonlinear mode decomposition with convolutional neural networks for fluid dynamics. *Journal of Fluid Mechanics*, 882, A13. <https://doi.org/10.1017/jfm.2019.822>
- Nakamura, T., Fukami, K., Hasegawa, K., Nabae, Y., & Fukagata, K. (2021). Convolutional neural network and long short-term memory based reduced order surrogate for minimal turbulent channel flow. *Physics of Fluids*, 33(2), 025116. <https://doi.org/10.1063/5.0039845>
- Noack, B. R., Morzynski, M., & Tadmor, G. (2011a). Reduced-order modelling for flow control. *New York: Springer*.
- Noack, B. R., Morzynski, M., & Tadmor, G. (2011b). *Reduced-order modelling for flow control* (Vol. 528). Springer Science & Business Media.
- Omata, N., & Shirayama, S. (2019). A novel method of low-dimensional representation for temporal behavior of flow fields using deep autoencoder. *AIP Advances*, 9(1), 015006. <https://doi.org/10.1063/1.5067313>
- Parish, E. J., & Carlberg, K. T. (2020). Time-series machine-learning error models for approximate solutions to parameterized dynamical systems. *Computer Methods in Applied Mechanics and Engineering*, 365, 112990. <https://doi.org/https://doi.org/10.1016/j.cma.2020.112990>
- Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., . . . Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in neural information processing systems* (Vol. 32). Curran Associates, Inc.
- Pathak, J., Hunt, B., Girvan, M., Lu, Z., & Ott, E. (2018). Model-free prediction of large spatiotemporally chaotic systems from data: A reservoir computing approach. *Phys. Rev. Lett.*, 120, 024102. <https://doi.org/10.1103/PhysRevLett.120.024102>
- Patton, A. (2013). Chapter 16 - copula methods for forecasting multivariate time series. In G. Elliott & A. Timmermann (Eds.), *Handbook of economic forecasting* (pp. 899–960, Vol. 2). Elsevier. <https://doi.org/https://doi.org/10.1016/B978-0-444-62731-5.00016-6>
- Pawar, S., Rahman, S. M., Vaddireddy, H., San, O., Rasheed, A., & Vedula, P. (2019). A deep learning enabler for nonintrusive reduced order modeling of fluid flows. *Physics of Fluids*, 31(8), 085101. <https://doi.org/10.1063/1.5113494>
- Penedo, G., Kydlíček, H., allal, L. B., Lozhkov, A., Mitchell, M., Raffel, C., Werra, L. V., & Wolf, T. (2024). The fineweb datasets: Decanting the web for the finest text data at scale. *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.
- Piomelli, U. (1999). Large-eddy simulation: Achievements and challenges. *Progress in Aerospace Sciences*, 35(4), 335–362. [https://doi.org/https://doi.org/10.1016/S0376-0421\(98\)00014-1](https://doi.org/https://doi.org/10.1016/S0376-0421(98)00014-1)
- Qin, T., Wu, K., & Xiu, D. (2019). Data driven governing equations approximation using deep neural networks. *Journal of Computational Physics*, 395, 620–635. <https://doi.org/https://doi.org/10.1016/j.jcp.2019.06.042>
- Quarteroni, A., Manzoni, A., & Negri, F. (2016). Reduced basis methods for partial differential equations. *Springer*.

- Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language models are unsupervised multitask learners.
- Rahman, S. M., Rasheed, A., & San, O. (2018). A hybrid analytics paradigm combining physics-based modeling and data-driven modeling to accelerate incompressible flow solvers. *Fluids*, 3(3). <https://doi.org/10.3390/fluids3030050>
- Raissi, M., Perdikaris, P., & Karniadakis, G. E. (2018). Multistep neural networks for data-driven discovery of nonlinear dynamical systems.
- Rapún, M.-L., Terragni, F., & Vega, J. M. (2015). Adaptive pod-based low-dimensional modeling supported by residual estimates. *International Journal for Numerical Methods in Engineering*, 104(9), 844–868. <https://doi.org/10.1002/nme.4947>
- Regazzoni, F., Dedè, L., & Quarteroni, A. (2019). Machine learning for fast and reliable solution of time-dependent differential equations. *Journal of Computational Physics*, 397, 108852. <https://doi.org/10.1016/j.jcp.2019.07.050>
- Reiss, J., Schulze, P., Sesterhenn, J., & Mehrmann, V. (2018). The shifted proper orthogonal decomposition: A mode decomposition for multiple transport phenomena. *SIAM Journal on Scientific Computing*, 40(3), A1322–A1344. <https://doi.org/10.1137/17M1140571>
- Rempfer, D. (2000). On low-dimensional galerkin models for fluid flow. *Theoretical and Computational Fluid Dynamics*, 14(2), 75–88. <https://doi.org/10.1007/s001620050131>
- Rodriguez-Fernandez, V., Carrasco, A., Cheng, J., Scharf, E., Siew, P. M., & Linares, R. (2024). Language models are spacecraft operators.
- Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In N. Navab, J. Hornegger, W. M. Wells, & A. F. Frangi (Eds.), *Medical image computing and computer-assisted intervention – miccai 2015* (pp. 234–241). Springer International Publishing.
- Rowley, C. W., Colonius, T., & Murray, R. M. (2004). Model reduction for compressible flows using pod and galerkin projection. *Physica D: Nonlinear Phenomena*, 189(1), 115–129. <https://doi.org/10.1016/j.physd.2003.03.001>
- Rowley, C. W., & Dawson, S. T. (2017). Model reduction for flow analysis and control. *Annual Review of Fluid Mechanics*, 49(Volume 49, 2017), 387–417. <https://doi.org/10.1146/annurev-fluid-010816-060042>
- Rühling Cachay, S., Zhao, B., Joren, H., & Yu, R. (2023). Dyffusion: A dynamics-informed diffusion model for spatiotemporal forecasting. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, & S. Levine (Eds.), *Advances in neural information processing systems* (pp. 45259–45287, Vol. 36). Curran Associates, Inc.
- San, O., & Iliescu, T. (2013). Proper orthogonal decomposition closure models for fluid flows: Burgers equation.
- San, O., & Maulik, R. (2018a). Extreme learning machine for reduced order modeling of turbulent geophysical flows. *Phys. Rev. E*, 97, 042322. <https://doi.org/10.1103/PhysRevE.97.042322>
- San, O., & Maulik, R. (2018b). Neural network closures for nonlinear model order reduction. *Advances in Computational Mathematics*, 44(6), 1717–1750. <https://doi.org/10.1007/s10444-018-9590-z>
- Schmid, P. (2010). Dynamic mode decomposition of numerical and experimental data. *J. Fluid Mech.*, 656, 5–28.

- Sengupta, A., Abadía-Heredia, R., Hetherington, A., Pérez, J. M., & Clainche, S. L. (2025). Hybrid machine learning models based on physical patterns to accelerate cfd simulations: A short guide on autoregressive models.
- Shi, X., Chen, Z., Wang, H., Yeung, D., Wong, W., & Woo, W. (2015). Convolutional lstm network: A machine learning approach for precipitation nowcasting.
- Shysheya, A., Diaconu, C., Bergamin, F., Perdikaris, P., Hernández-Lobato, J. M., Turner, R. E., & Mathieu, E. (2024). On conditional diffusion models for pde simulations.
- Sieber, M., Paschereit, C. O., & Oberleithner, K. (2016). Spectral proper orthogonal decomposition. *Journal of Fluid Mechanics*, 792, 798–828. <https://doi.org/10.1017/jfm.2016.103>
- Sirovich, L. (1987). Turbulence and the dynamics of coherent structures. Parts I–III. *Quarterly of applied mathematics*, 45(3), 561–571.
- Smith, M. S. (2023). Implicit copulas: An overview. *Econometrics and Statistics*, 28, 81–104. <https://doi.org/https://doi.org/10.1016/j.ecosta.2021.12.002>
- Solomon, T. H. (2009). Non-linear fluid flow, pattern formation, mixing and turbulence. In R. A. Meyers (Ed.), *Encyclopedia of complexity and systems science* (pp. 6195–6206). Springer New York. https://doi.org/10.1007/978-0-387-30440-3_362
- Song, Y., & Ermon, S. (2020). Improved techniques for training score-based generative models. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, & H. Lin (Eds.), *Advances in neural information processing systems* (pp. 12438–12448, Vol. 33). Curran Associates, Inc.
- Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., & Poole, B. (2021). Score-based generative modeling through stochastic differential equations.
- Stamatelopoulos, S., & Sapsis, T. P. (2025). Can diffusion models capture extreme event statistics? *Computer Methods in Applied Mechanics and Engineering*, 435, 117589. <https://doi.org/https://doi.org/10.1016/j.cma.2024.117589>
- Sun, L., Han, X., Gao, H., Wang, J.-X., & Liu, L. (2023). Unifying predictions of deterministic and stochastic physics in mesh-reduced space with sequential flow generative model. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, & S. Levine (Eds.), *Advances in neural information processing systems* (pp. 60636–60660, Vol. 36). Curran Associates, Inc.
- Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction*. A Bradford Book.
- Tadmor, E. (2012). A review of numerical methods for nonlinear partial differential equations. *Bull. Amer. Math. Soc.*, 49, 507–554. <https://doi.org/https://doi.org/10.1090/S0273-0979-2012-01379-4>
- Temirchev, P., Simonov, M., Kostoev, R., Burnaev, E., Oseledets, I., Akhmetov, A., Margarit, A., Sitnikov, A., & Koroteev, D. (2020). Deep neural networks predicting oil movement in a development unit. *Journal of Petroleum Science and Engineering*, 184, 106513. <https://doi.org/https://doi.org/10.1016/j.petrol.2019.106513>
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, & R. Garnett (Eds.), *Advances in neural information processing systems* (Vol. 30). Curran Associates, Inc.

- Vega, J. M., & Le Clainche, S. (2021). *Higher order dynamic mode decomposition and its applications*. Academic Press. <https://doi.org/10.1016/B978-0-12-819743-1.00007-0>
- Vlachas, P. R., Byeon, W., Wan, Z. Y., Sapsis, T. P., & Koumoutsakos, P. (2018). Data-driven forecasting of high-dimensional chaotic systems with long short-term memory networks. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 474(2213), 20170844. <https://doi.org/10.1098/rspa.2017.0844>
- Wang, Y., Zeng, Y., Deiterding, R., & Liang, J. (2025). An efficient gpu-accelerated adaptive mesh refinement framework for high-fidelity compressible reactive flows modeling.
- Wang, Z., Akhtar, I., Borggaard, J., & Iliescu, T. (2012). Proper orthogonal decomposition closure models for turbulent flows: A numerical comparison. *Computer Methods in Applied Mechanics and Engineering*, 237-240, 10–26. <https://doi.org/10.1016/j.cma.2012.04.015>
- Wu, P. L., Gao, H. S., Wang, Q., & Wang, P. Y. (2022). A novel forecast framework for unsteady flows based on a convolutional neural network. *Physics of Fluids*, 34(11), 113611. <https://doi.org/10.1063/5.0122271>
- Xiao, D., Heaney, C., Mottet, L., Fang, F., Lin, W., Navon, I., Guo, Y., Matar, O., Robins, A., & Pain, C. (2019). A reduced order model for turbulent flows in the urban environment using machine learning. *Building and Environment*, 148, 323–337. <https://doi.org/10.1016/j.buildenv.2018.10.035>
- Xie, X., Zhang, G., & Webster, C. G. (2019). Non-intrusive inference reduced order model for fluids using deep multistep neural network. *Mathematics*, 7(8). <https://doi.org/10.3390/math7080757>
- Xu, L., Zhou, G., Zhao, F., Guo, Z., & Zhang, K. (2022). A data-driven reduced order modeling for fluid flow analysis based on series forecasting intelligent algorithm. *IEEE Access*, 10, 60163–60176. <https://doi.org/10.1109/ACCESS.2022.3177223>
- Yeo, K., & Melnyk, I. (2019). Deep learning algorithm for data-driven simulation of noisy dynamical system. *Journal of Computational Physics*, 376, 1212–1231. <https://doi.org/10.1016/j.jcp.2018.10.024>
- Yeung, P., Ravikumar, K., Nichols, S., & Uma-Vaideswaran, R. (2025). Gpu-enabled extreme-scale turbulence simulations: Fourier pseudo-spectral algorithms at the exascale using openmp offloading. *Computer Physics Communications*, 306, 109364. <https://doi.org/10.1016/j.cpc.2024.109364>
- Zabusky, N. J., & Kruskal, M. D. (1965). Interaction of "solitons" in a collisionless plasma and the recurrence of initial states. *Phys. Rev. Lett.*, 15, 240–243. <https://doi.org/10.1103/PhysRevLett.15.240>
- Zawawi, M. H., Saleha, A., Salwa, A., Hassan, N. H., Zahari, N. M., Ramli, M. Z., & Muda, Z. C. (2018). A review: Fundamentals of computational fluid dynamics (cfd). *AIP Conference Proceedings*, 2030(1), 020252. <https://doi.org/10.1063/1.5066893>

Appendix A

Data Preprocessing

All datasets utilized in this thesis are represented as spatio-temporal tensors composed of S snapshots, denoted as $\mathbf{v}_s \equiv \mathbf{v}(t_s)$, where $\mathbf{v}(t_s)$ corresponds to the flow field at the s -th time instant. For notational convenience, these snapshots are assembled into the following snapshot tensor,

$$\mathbf{v} = [\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_s, \dots, \mathbf{v}_S]. \quad (\text{A.1})$$

Each $\mathbf{v}(t_s)$ is itself a tensor, comprising both the velocity components and the spatial dimensions. In this work, the components dimension corresponds to the velocity components, typically streamwise, wall-normal and spanwise. For example, one of the datasets used is two-dimensional in space and contains only a single velocity component.

This dataset represents the velocity field $\mathbf{v}$ in a Cartesian coordinate system, with spatial dimensions denoted as $J_2 \times J_3$, such that,

$$\begin{aligned} \mathbf{v}_s &= \mathbf{v}(x_{j_2}, y_{j_3}, t_s), \quad j_2 \in \{1, \dots, J_2\}, \\ j_3 &\in \{1, \dots, J_3\}, \quad s \in \{1, \dots, S\}. \end{aligned} \quad (\text{A.2})$$

Accordingly, the snapshot tensor $\mathring{\mathbf{D}}$ can be reorganized as a fourth-order tensor of dimensions $J_1 \times J_2 \times J_3 \times S$, where $J_1 = 1$ denotes the number of velocity components in the dataset.

$$\mathring{D}_{1j_2j_3k} = \mathbf{v}_k^1 = \mathbf{v}^1(x_{j_2}, y_{j_3}, t_k). \quad (\text{A.3})$$

Another dataset used in this thesis is three-dimensional in space and includes two velocity components. It can be represented using the same notation, where $\mathbf{v}_s = \mathbf{v}(x_{j_2}, y_{j_3}, z_{j_4}, t_s)$, and the full dataset is structured as a fifth-order tensor $\mathbf{v} \in \mathbb{R}^{J_1 \times J_2 \times J_3 \times J_4 \times S}$, with $J_1 = 2$ representing the number of velocity components.

$$\mathring{D}_{1j_2j_3j_4s} = \mathbf{v}_s^1 = \mathbf{v}^1(x_{j_2}, y_{j_3}, z_{j_4}, t_s), \quad (\text{A.4})$$

$$\mathring{D}_{2j_2j_3j_4s} = \mathbf{v}_s^2 = \mathbf{v}^2(x_{j_2}, y_{j_3}, z_{j_4}, t_s). \quad (\text{A.5})$$

Note that J_1 denotes the number of velocity components, while J_2 , J_3 , and J_4 correspond to the spatial dimensions in the x , y , and z directions, respectively. The dimension S represents the temporal extent of the dataset.

The following two sections describe, first, a data augmentation process that leverages the three spatial dimensions of a dataset, and second, the rolling window technique employed to prepare the time series for training the forecasting models.

A.1 Data augmentation

Datasets with three dimensions in space pose significant challenges for machine learning models, particularly when tasked with capturing both spatial and temporal correlations under conditions of limited data availability, an issue central to this thesis. This challenge has been previously acknowledged in the literature (Nakamura et al., 2021), with most solutions relying on access to extensive training datasets to compensate for the inherent complexity.

To mitigate these limitations, a data augmentation strategy is implemented. The complete tensor $\mathring{D}$ (as defined in eq. (A.1)) is systematically partitioned into J_4 non-overlapping sub-tensors, as detailed below,

$$\mathring{D}_{j_1j_2j_31s} = \mathbf{v}(c_{j_1}, x_{j_2}, y_{j_3}, z_1, t_s), \quad (\text{A.6})$$

$$\mathring{D}_{j_1j_2j_32s} = \mathbf{v}(c_{j_1}, x_{j_2}, y_{j_3}, z_2, t_s), \quad (\text{A.7})$$

$$\vdots \quad (\text{A.8})$$

$$\mathring{D}_{j_1j_2j_3J_4s} = \mathbf{v}(c_{j_1}, x_{j_2}, y_{j_3}, z_{J_4}, t_s). \quad (\text{A.9})$$

As illustrated in Eqs. A.4 and A.5, the elements along the z -dimension of the tensor $\mathring{D}$ can be treated as independent entities when dealing with three-dimensional datasets. This permits the decomposition of the original dataset into J_4 distinct two-dimensional datasets (Eqs. A.6–A.9), each corresponding to a separate time series, thereby reframing the original three-dimensional dataset as a collection of independent temporal sequences.

This approach enhances the performance of forecasting models by effectively increasing the number of training samples by a factor of J_4 . Nonetheless, this partitioning strategy may lead to a loss of information in the z dimension (spanwise), as the spatial continuity across this dimension is no longer preserved.

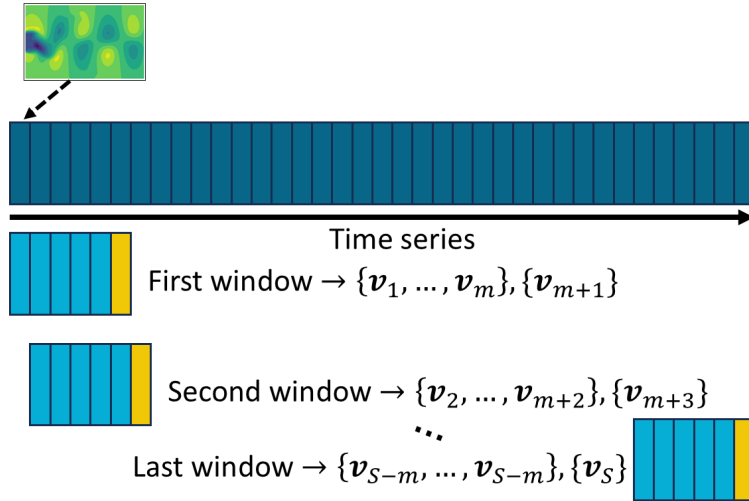


Figure A.1: Sliding window approach applied to the time series with a predefined stride. Each window comprises a sequence of preceding snapshots, serving as inputs to the forecasting model, and a subsequent snapshot, which constitutes the prediction target. In this illustrative example, m prior snapshots are utilized to forecast the immediate next state.

A.2 Rolling window

In contrast to the data augmentation strategy described above, which is specifically designed for three-dimensional datasets and aims to expand the number of training samples, the rolling window technique is a general purpose method for preparing time series data for forecasting and is applicable to datasets of any dimensionality.

Time series forecasting necessitates a clear specification of the input sequence (historical observations) and the prediction target (future values). The rolling window technique allows this process by systematically traversing the time series with a predefined stride, generating overlapping windows that include both the input sequence and the corresponding target value. Applied independently to each time series, this technique yields a collection of input–target pairs suitable for training forecasting models. Fig. A.1 illustrates this procedure for a representative time series.

Prior to model training, all windows are normalized using different scaling methods as shown in Chapter 2. This normalization enhance training stability and performance by mitigating issues such as gradient vanishing or explosion. Architectural strategies are also employed to address these challenges. For instance in Chapter 3, the residual autoencoder incorporates skip connections explicitly designed to stabilize gradient flow. Similarly, both the Variational Autoencoder and the POD-DL models utilize LSTM layers to preserve long-range dependencies, countering the vanishing gradient problem characteristic of recurrent neural networks. Furthermore, careful weight initialization was adopted across all models to further minimize the risk of gradient instability.

Appendix B

Computation of the probability of prediction error (PPE)

Algorithm 1 presents the pseudocode used to compute the probability of achieving an absolute prediction error less than or equal to a specified threshold (err_thresh). This is evaluated by comparing reference snapshots (gt_ten) with their corresponding predictions ($pred_ten$), as defined in the probability of prediction error (PPE) metric introduced in eq. (4.2).

Algorithm 1 Compute probabilistic prediction error (PPE)

```
1: function COMPUTE_PPE( $err\_thresh$ ,  $bin\_width$ ,  $gt\_ten$ ,  $pred\_ten$ )
2:    $diff \leftarrow$  absolute value of ( $gt\_ten - pred\_ten$ )
3:    $max\_diff \leftarrow$  maximum value in  $diff$ 
4:    $bin\_edges \leftarrow$  array from 0.0 to  $max\_diff + bin\_width$  with step  $bin\_width$ 
5:   ( $p\_hist$ ,  $\_$ )  $\leftarrow$  histogram of  $diff$  using  $bin\_edges$ , normalized to density
6:    $p \leftarrow p\_hist \times bin\_width$ 
7:    $index \leftarrow$  integer part of  $\frac{err\_thresh}{bin\_width}$ 
8:   return sum of  $p$  from bin 0 to bin ( $index - 1$ )
9: end function
```

Appendix C

Model architectures and analysis of autoencoder latent spaces

This appendix presents tables detailing the architectures of the models introduced in Chapter 2 and Chapter 3: the POD-DL model (Table C.1), the residual autoencoder (RAE) (Tables C.2 – C.4), and the variational autoencoder (VAE) (Table C.5 and Table C.6). It is important to note that the architectures of the autoencoders are dataset-dependent. The configurations provided in these tables correspond to the flow past a cylinder under laminar conditions (Le Clainche et al., 2018), as the network architecture is intrinsically linked to the spatial resolution of the input snapshots.

Figures C.1–C.3 display some latent variables produced by the encoders of the RAE and VAE models, respectively, when applied to the laminar flow dataset. These visualizations reveal distinct spatial structures within the flow dynamics, indicating that the encoders of both models effectively decompose the flow into its underlying dynamic components.

It is important to note that the VAE implementation discussed in this thesis is limited to training on a single velocity component at a time, in contrast to the RAE, which can accommodate multiple velocity components simultaneously. Accordingly, Fig. C.1 presents representative latent variables from the RAE encoder corresponding to both the streamwise and wall-normal velocity components. In contrast, Figures C.2 and C.3 display representative latent variables from the VAE encoder for the same velocity components, with separate training performed for each component.

Table C.1: Architecture details in the POD-DL model. Unlike the other two models that receive snapshots as inputs, the POD-DL receive vectors representing the retained POD modes. In the case of the laminar flow is 20, as explained in Chapter 3. The term $(6\times)$ in the second and third layer, mean that 6 layers with the same configuration are trained in parallel. The architecture used in Chapter 2 is structurally identical, but employs a single layer instead of six, as it computes a single prediction.

Layer	Layer details	# Neurons	Activation function	Dimension
0	Input	N	-	10×20
1	LSTM	$6 * 100$	ReLU	1×600
2	Reshape	-	-	$6 \times 1 \times 100$
3	$(6\times)$ FC	$(6\times) 80$	ReLU	$6 \times 1 \times 80$
4	$(6\times)$ FC	$(6\times) 20$	Linear	$6 \times 1 \times 20$

Table C.2: Encoder architecture of the residual autoencoder (RAE) for input snapshots corresponding to the laminar flow dataset. Note that this architecture is resolution-dependent and may vary with the size of the input snapshots. Specifically, the number of Identity Blocks (Identity B.) and Convolutional Blocks (Conv B.) can be adjusted—added or removed—based on the spatial resolution of the data.

# Layer	Layer details	Kernel size	Stride	Padding	Activation	# Params.	Dimension
0	Input	-	-	-	-	-	$10 \times 100 \times 40 \times 2$
1	Conv 2D	3×3	2×1	Valid	ReLU	448	$10 \times 49 \times 38 \times 16$
1	LayerNorm.	-	-	-	-	32	$10 \times 49 \times 38 \times 16$
2	Conv 2D	3×3	1×1	Valid	ReLU	4640	$10 \times 47 \times 36 \times 32$
2	LayerNorm.	-	-	-	-	64	$10 \times 47 \times 36 \times 32$
3	Identity B.	3×3	1×1	Same	ReLU	18560	$10 \times 47 \times 36 \times 32$
4	Identity B.	3×3	1×1	Same	ReLU	18560	$10 \times 47 \times 36 \times 32$
5	Identity B.	3×3	1×1	Same	ReLU	18560	$10 \times 47 \times 36 \times 32$
6	Conv B.	3×3	2×2	Valid	ReLU	57600	$10 \times 24 \times 18 \times 64$
7	Identity B.	3×3	1×1	Same	ReLU	73984	$10 \times 24 \times 18 \times 64$
8	Identity B.	3×3	1×1	Same	ReLU	73984	$10 \times 24 \times 18 \times 64$
9	Identity B.	3×3	1×1	Same	ReLU	73984	$10 \times 24 \times 18 \times 64$

Table C.3: Same as Table C.2 for the ConvLSTM layer in the RAE.

# Layer	Layer details	Kernel size	Stride	Padding	Activation	# Params.	Dimension
0	Input	-	-	-	-	-	$10 \times 24 \times 18 \times 64$
1	ConvLSTM	3×3	1×1	Same	Tanh/Sigmoid	884736	$1 \times 24 \times 18 \times 128$
1	LayerNorm.	-	-	-	-	256	$1 \times 24 \times 18 \times 128$

Table C.4: Same as Table C.2 for the decoder in the RAE. As in the encoder, the number of transpose convolutional layers depends on the size of the input snapshots.

# Layer	Layer details	Kernel size	Stride	Padding	Activation	# Params.	Dimension
0	Input	-	-	-	-	-	$1 \times 24 \times 18 \times 128$
1	Conv 2D T.	3×3	2×2	Valid	ReLU	73728	$1 \times 49 \times 37 \times 64$
1	LayerNorm.	-	-	-	-	128	$1 \times 49 \times 37 \times 64$
2	Conv 2D T.	4×4	2×1	Valid	ReLU	32768	$1 \times 100 \times 40 \times 32$
2	LayerNorm.	-	-	-	-	64	$1 \times 100 \times 40 \times 32$
3	Conv 2D	1×1	1×1	Valid	Linear	96	$1 \times 100 \times 40 \times 2$

Table C.5: Encoder architecture of the variational autoencoder (VAE) for input snapshots corresponding to the laminar flow dataset. Note that this architecture is resolution-dependent and may vary with the size of the input snapshots. Specifically, the number of Identity Blocks (Identity B.) and Convolutional Blocks (Conv B.) can be adjusted—added or removed—based on the spatial resolution of the data.

# Layer	Layer details	Kernel size	Stride	Padding	Activation	# Params.	Dimension
0	Input	-	-	-	-	-	$10 \times 100 \times 40 \times 1$
1	Conv 2D	3×3	2×1	Valid	ReLU	400	$10 \times 94 \times 34 \times 8$
1	LayerNorm.	-	-	-	-	16	$10 \times 94 \times 34 \times 8$
2	Conv 2D	3×3	1×1	Valid	ReLU	1168	$10 \times 46 \times 16 \times 16$
2	LayerNorm.	-	-	-	-	32	$10 \times 46 \times 16 \times 16$
3	Conv 2D	3×3	1×1	Valid	ReLU	4640	$10 \times 22 \times 14 \times 32$
3	LayerNorm.	-	-	-	-	64	$10 \times 22 \times 14 \times 32$
4	ConvLSTM	3×3	2×1	Valid	ReLU	221440	$1 \times 11 \times 14 \times 64$
5	Flatten	-	-	-	-	-	1×9856
6	Dense	-	-	-	-	3449950	1×350
7	Posterior dist.	-	-	-	-	-	1×25

Table C.6: Same as Table C.5 for the decoder in the VAE. As in the encoder, the number of transpose convolutional layers depends on the size of the input snapshots.

# Layer	Layer details	Kernel size	Stride	Padding	Activation	# Params.	Dimension
0	Input	-	-	-	-	-	1×25
1	Dense	-	-	-	-	256256	1×9856
2	Reshape	-	-	-	-	-	$1 \times 11 \times 14 \times 64$
3	Conv 2D T.	3×3	2×2	Valid	ReLU	73728	$1 \times 23 \times 16 \times 64$
3	LayerNorm.	-	-	-	-	128	$1 \times 23 \times 16 \times 64$
4	Conv 2D T.	4×4	2×1	Valid	ReLU	32768	$1 \times 47 \times 18 \times 32$
4	LayerNorm.	-	-	-	-	64	$1 \times 47 \times 18 \times 32$
5	Conv 2D T.	3×3	2×2	Valid	ReLU	73728	$1 \times 96 \times 37 \times 16$
5	LayerNorm.	-	-	-	-	128	$1 \times 96 \times 37 \times 16$
6	Conv 2D T.	3×3	2×2	Valid	ReLU	73728	$1 \times 100 \times 40 \times 8$
6	LayerNorm.	-	-	-	-	128	$1 \times 100 \times 40 \times 8$
7	Likelihood dist.	-	-	-	-	353565	$1 \times 100 \times 40 \times 1$

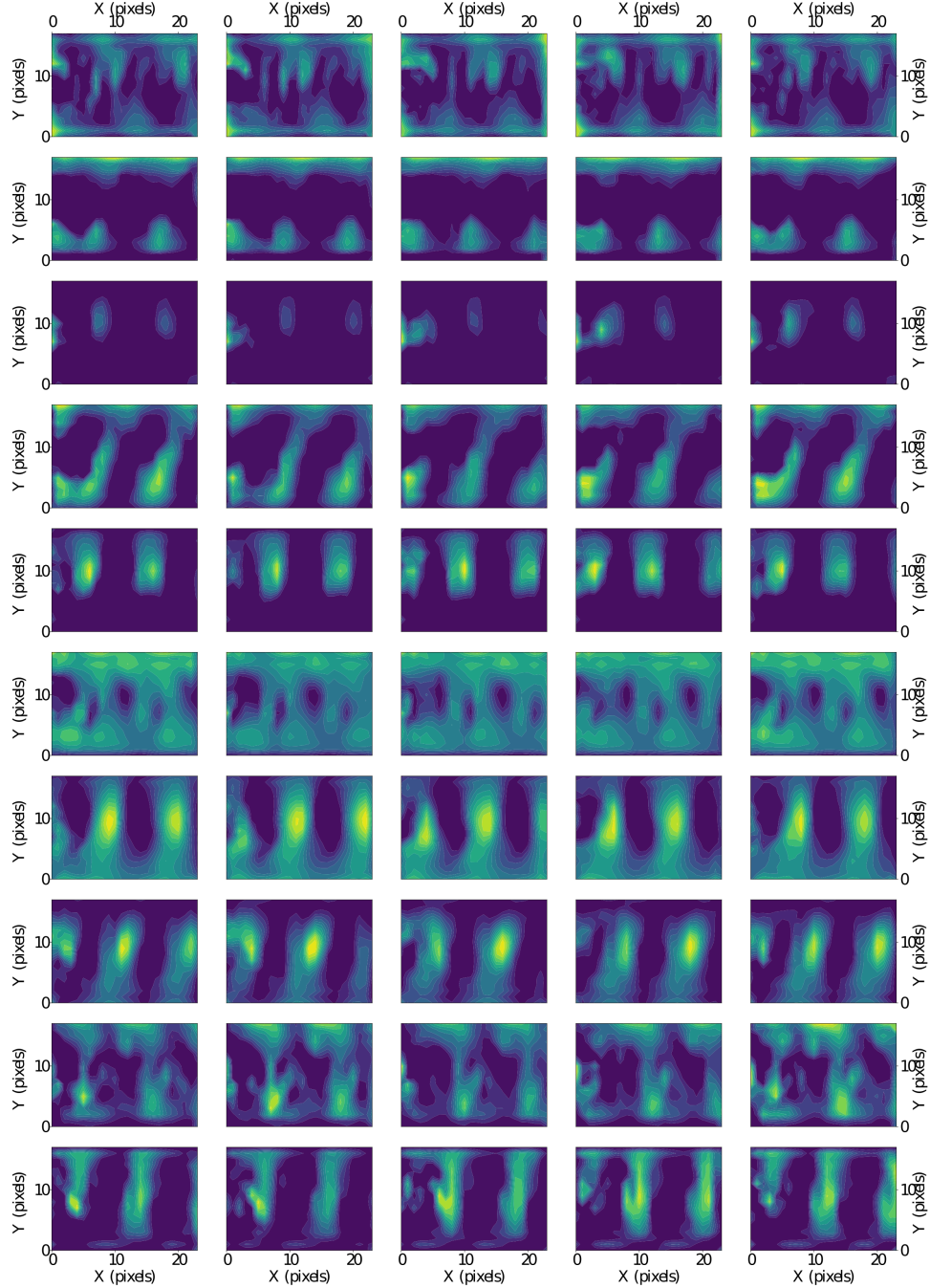


Figure C.1: Snapshots of selected latent variables extracted from the encoder of the RAE, based on input data comprising the two velocity components of the flow past a circular cylinder at laminar regime (Le Clainche et al., 2018). For this dataset, the encoder receives as input a sequence of 10 prior snapshots with dimensions $[10 \times 100 \times 40 \times 3]$ and outputs a latent tensor of size $[10 \times 24 \times 18 \times 64]$. The figure displays channels 1, 2, 3, 4, 5, 10, 20, 35, 55, and 64 from top to bottom. The temporal evolution of each channel is shown from left to right over five consecutive time steps. All dimensions shown are in pixels, corresponding to the spatial resolution of the latent variables.

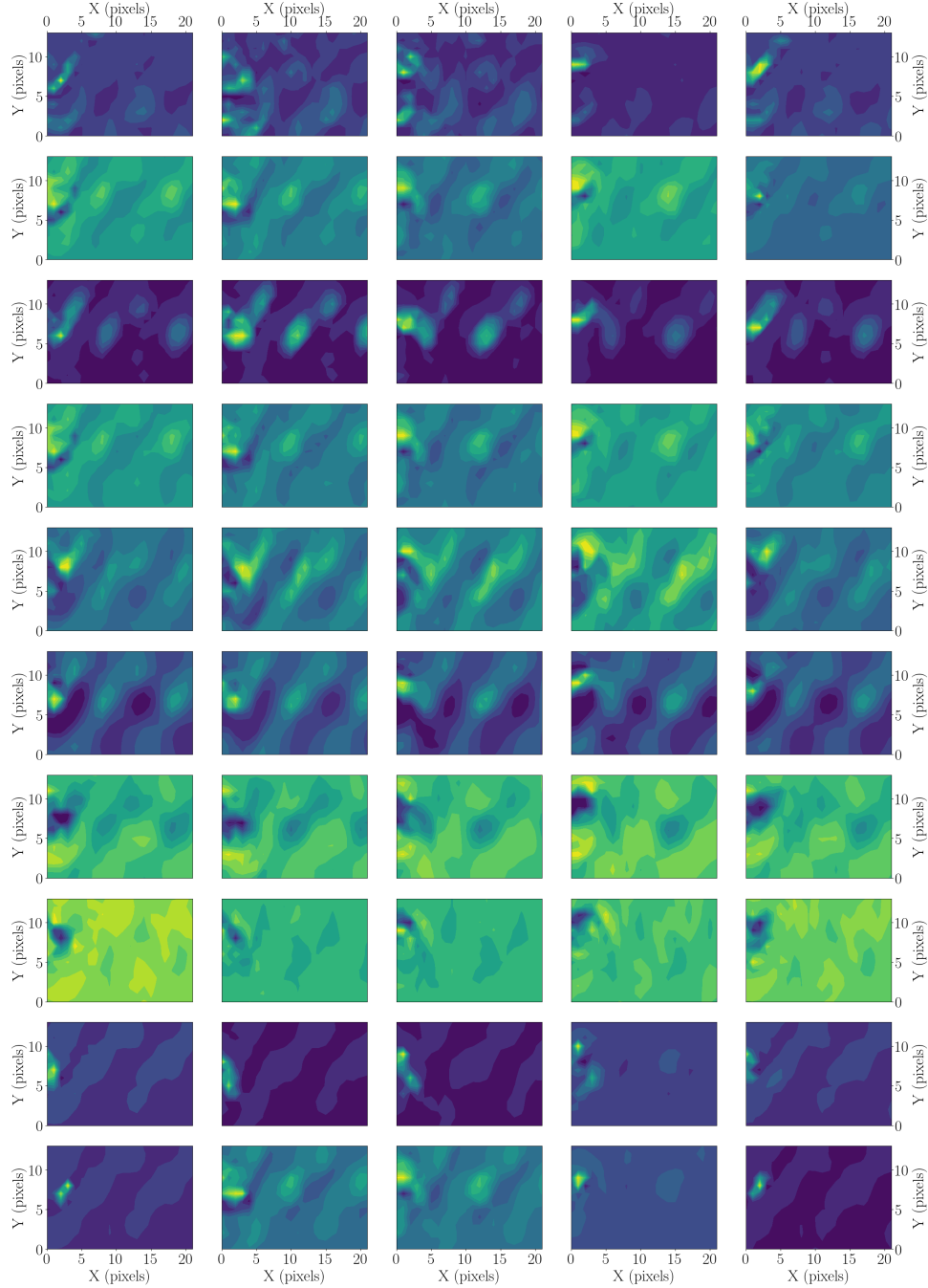


Figure C.2: Snapshots of selected latent variables extracted from the encoder of the VAE, based on input data comprising the streamwise velocity component of the flow past a circular cylinder at laminar regime (Le Clainche et al., 2018). For this dataset, the encoder receives as input a sequence of 10 prior snapshots with dimensions $[10 \times 100 \times 40 \times 3]$ and outputs a latent tensor of size $[10 \times 24 \times 18 \times 64]$. The figure displays channels 1, 2, 3, 4, 5, 6, 9, 16, 25, and 32 from top to bottom. The temporal evolution of each channel is shown from left to right over five consecutive time steps. All dimensions shown are in pixels, corresponding to the spatial resolution of the latent variables.

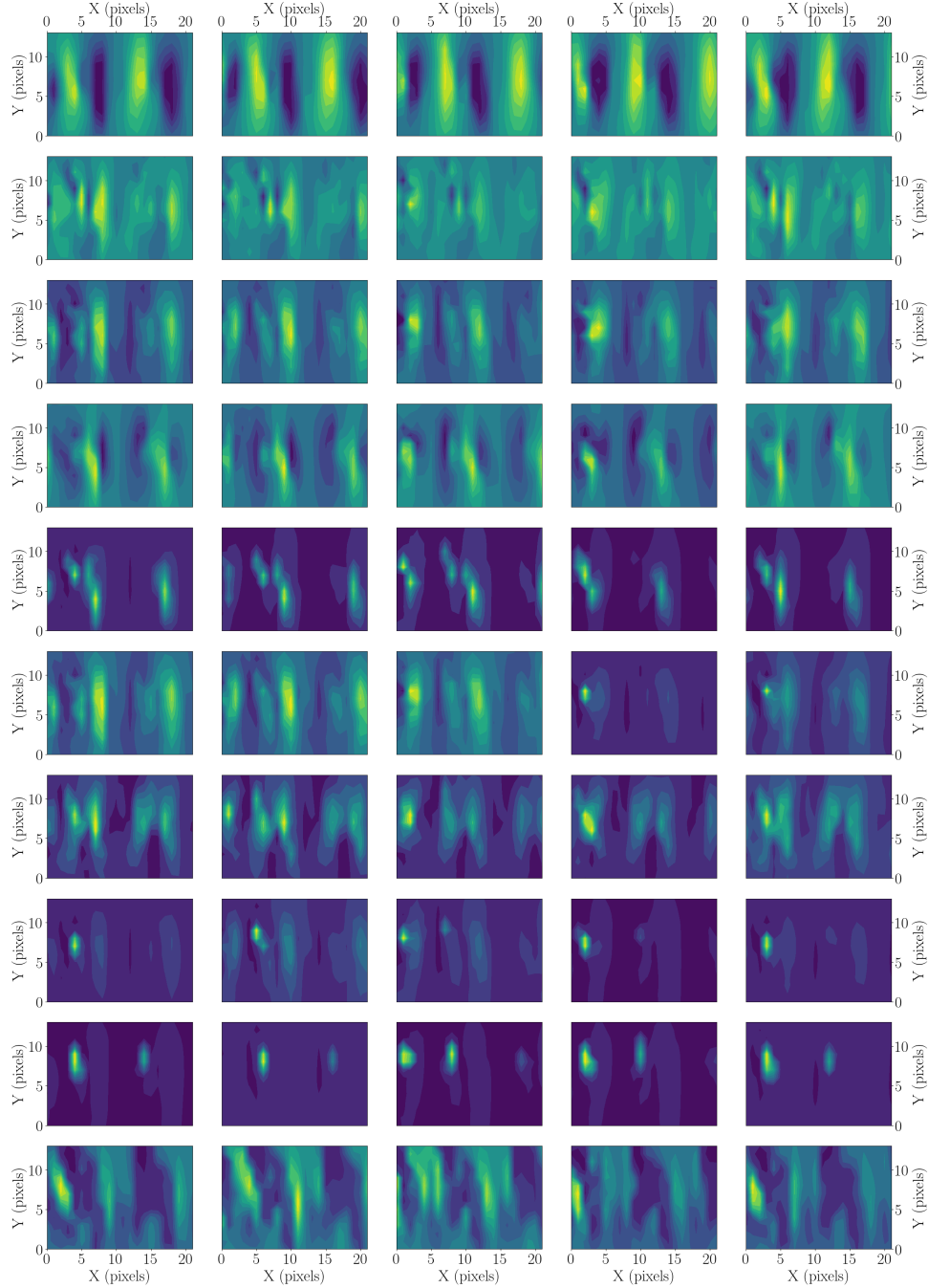


Figure C.3: Snapshots of selected latent variables extracted from the encoder of the VAE, based on input data comprising the normal velocity component of the flow past a circular cylinder at laminar regime (Le Clainche et al., 2018). For this dataset, the encoder receives as input a sequence of 10 prior snapshots with dimensions $[10 \times 100 \times 40 \times 3]$ and outputs a latent tensor of size $[10 \times 24 \times 18 \times 64]$. The figure displays channels 1, 2, 3, 4, 5, 6, 9, 16, 25, and 32 from top to bottom. The temporal evolution of each channel is shown from left to right over five consecutive time steps. All dimensions shown are in pixels, corresponding to the spatial resolution of the latent variables.