

Reduced-Order Modeling of Turbulent Boundary Layer

Dhyan G. (AE22B030), Department of Aerospace Engineering, IIT Madras

Abstract—This work develops reduced-order models for a transitional/turbulent boundary layer using DNS data from the Johns Hopkins Turbulence Database (JHTDB) [1]. Three modeling approaches are considered: Dynamic Mode Decomposition, Operator Inference, and Sparse Identification of Nonlinear Dynamics. The Reynolds number range used is $Re \in [400000, 440000]$. The ROMs are evaluated for reconstruction accuracy, future-state prediction capability, and physical interpretability. Mean-flow statistics, autocorrelation behavior, turbulent scales, and near-wall quantities such as skin friction are analyzed. An Energy-Scaled OpInf framework is further introduced to recover physically consistent fluctuation energy. All methods are compared on identical POD-projected datasets containing 50 modes. Results show that OpInf and ES-OpInf significantly outperform baseline DMD, while SINDy provides interpretable but highly sensitive governing equations.

I. INTRODUCTION

Turbulent boundary layers are extremely costly to simulate using Direct Numerical Simulation with coherent structures in sufficiently high clarity for different applications that require levels of accuracy, making Reduced-Order Modeling essential for prediction with lesser computational cost. High-fidelity DNS data from JHTDB's `transition.bl` dataset paves the way for data-driven learning of the dominant spatial and temporal structures.

In this work, we focus on two ROM techniques and one equation discovery technique:

- **Dynamic Mode Decomposition (DMD)**
- **Operator Inference (OpInf)**
- **SINDy**

A fixed set of 50 POD modes is used for projection to obtain modal coefficients for all ROMs. For the first two methods, the data was split in the ratio 4:1 as train:test.

II. DATASET DESCRIPTION

The `transition.bl` dataset contains unsteady transitional boundary layer data with velocity components (u, v, w) and pressure p over a 3D grid:

- $x \in [500, 550]$, $y \in [0.0036, 7.0]$, $z = 120$ all the ranges were equally divided. To also include some inner boundary characteristics, the sampling frequency of the y grid was increased, and it was also sampled in the nearest measured boundary layer location.
- Grid size: i) 36×113 , ii) 35×113 and both of the grids were concatenated to get 71×113 grid. This was done due to the maximum number of gridpoints (i.e., 4096) which could be accessed at a time in the database.

- Time range: $t = 100$ to 200 with $\Delta t = 0.5$. The timestep range was chosen in accordance to the autocorrelation plots obtained in 5 different y locations as in 2.

The quantity taken to study the effect on the near-wall behaviour was the skin friction coefficient.

$$C_f = \frac{\tau_w}{0.5\rho U_\infty^2}, \quad \tau_w = \mu \left(\frac{\partial u}{\partial y} \right)_{y=0} \quad (1)$$

The general data extracted for studying the contours were u or U_x which can be used to derive the quantities studied later. The ensemble average data is given in JHTDB [1] and is used in autocorrelation and analysis of the data.

A. Governing Equations

For unsteady incompressible turbulent boundary layers, the mean RANS equations are:

$$\begin{aligned} \frac{\partial \bar{u}_i}{\partial x_i} &= 0, \\ \frac{\partial \bar{u}_i}{\partial t} + \bar{u}_j \frac{\partial \bar{u}_i}{\partial x_j} &= -\frac{1}{\rho} \frac{\partial \bar{p}}{\partial x_i} + \nu \frac{\partial^2 \bar{u}_i}{\partial x_j^2} - \frac{\partial \bar{u}_i \bar{u}_j}{\partial x_j}. \end{aligned}$$

Simplified 2D Boundary-Layer Form (Mean Flow):

$$\begin{aligned} \frac{\partial \bar{u}}{\partial x} + \frac{\partial \bar{v}}{\partial y} &= 0, \\ \frac{\partial \bar{u}}{\partial t} + \bar{u} \frac{\partial \bar{u}}{\partial x} + \bar{v} \frac{\partial \bar{u}}{\partial y} &= -\frac{1}{\rho} \frac{d \bar{p}}{dx} + \nu \frac{\partial^2 \bar{u}}{\partial y^2} - \frac{\partial \bar{u} \bar{v}'}{\partial y}. \end{aligned}$$

Boundary Conditions:

$$\begin{aligned} y = 0 : \quad \bar{u} &= 0, \bar{v} = 0 \quad (\text{no-slip}) \\ y \rightarrow \infty : \quad \bar{u} &\rightarrow U_\infty \end{aligned}$$

B. Length scales

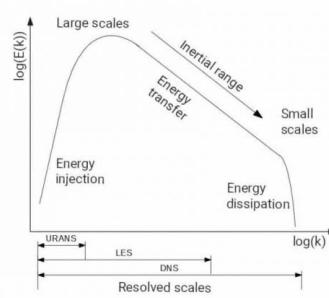


Fig. 1: Turbulent length scales [2]

The length scales were visually analysed and confirmed to be present in each timestep. As shown in the plot above, there are three scales of eddies: productive scales, inertial scales, and dissipative scales. ROMs can only capture and predict larger scales like productive and large inertial, but resolving dissipative scales require high fidelity models. Even DNS might not fully capture the dissipative scales with the resolution possible with today's supercomputers.

C. Interpretation of Autocorrelation and Choice of Sampling Timestep

Figure 2 shows the temporal autocorrelation of the streamwise velocity fluctuations at several wall-normal locations. The key observation is the presence of a secondary correlation peak at $t \approx 100$, indicating a dominant large-scale temporal structure in the flow.

The original JHTDB [1] sampling interval of $\Delta t = 0.25$ s provides 200 snapshots, but this time range(i.e.,100s-150s) is insufficient for an entire cycle of the large-scale motion visible in the autocorrelation curve. To ensure that at least one dominant “period” of the dynamics is represented within the available data window, a coarser sampling interval was selected.

Also, the correlations go to zero, indicating statistical stationarity [3], which was confirmed in the Turbulence Database.

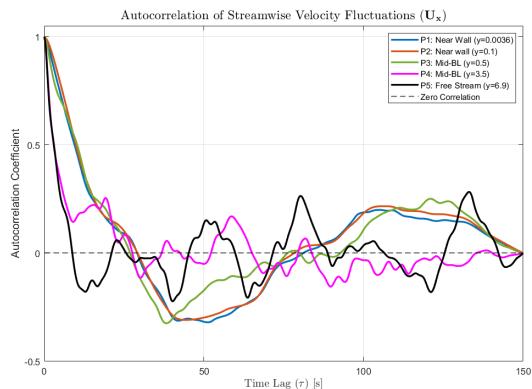


Fig. 2: Autocorrelation of u_x at selected wall-normal locations.

III. DYNAMIC MODE DECOMPOSITION

Dynamic Mode Decomposition provides a linear approximation of a high-dimensional system from measurements. Given snapshot matrices

$$\mathbf{X}_1 = [\mathbf{x}_1, \mathbf{x}_2, \dots], \quad \mathbf{X}_2 = [\mathbf{x}_2, \mathbf{x}_3, \dots],$$

with sampling interval Δt , an economy-size SVD is applied:

$$\mathbf{X}_1 = \mathbf{U} \mathbf{S} \mathbf{V}^\top, \quad (2)$$

dimensionality reduction:

$$\mathbf{U}_r = \mathbf{U}(:, 1:r), \quad \mathbf{S}_r = \mathbf{S}(1:r, 1:r), \quad \mathbf{V}_r = \mathbf{V}(:, 1:r). \quad (3)$$

The reduced Koopman operator is calculated as

$$\tilde{\mathbf{A}} = \mathbf{U}_r^\top \mathbf{X}_2 \mathbf{V}_r \mathbf{S}_r^{-1}. \quad (4)$$

Eigen-decomposition gives

$$\tilde{\mathbf{A}} \mathbf{W}_r = \mathbf{W}_r \mathbf{D}_r, \quad (5)$$

DMD spatial modes [4] are reconstructed as

$$\Phi_{\text{DMD}} = \mathbf{X}_2 \mathbf{V}_r \mathbf{S}_r^{-1} \mathbf{W}_r. \quad (6)$$

A. Two Operator-Inference Formulations

DMD operators may be inferred in two different ways. These inference formulations are distinct from the methods used later to predict the system evolution.

1) *Continuous-Time Operator Inference*: Assuming a latent linear ODE

$$\dot{\mathbf{x}} = A_c \mathbf{x}, \quad (7)$$

the continuous-time eigenvalues are obtained from the DMD eigenvalues λ_i via

$$\omega_i = \frac{1}{\Delta t} \log(\lambda_i), \quad (8)$$

leading to the continuous operator

$$A_c = \mathbf{W}_r \Omega \mathbf{W}_r^{-1}, \quad \Omega = \text{diag}(\omega_i). \quad (9)$$

2) *Discrete-Time Operator Inference*: Without assuming an ODE, the optimal discrete-time linear map is

$$A_d = \mathbf{X}_2 \mathbf{X}_1^\dagger, \quad (10)$$

giving the discrete dynamical system

$$\mathbf{x}_{k+1} = A_d \mathbf{x}_k. \quad (11)$$

B. Three Prediction Methods

After inferring either A_c or A_d , the system may be propagated in time using one of three approaches.

1) *Numerical ODE Integration*: If A_c is available, a continuous-time evolution may be computed via

$$\dot{\mathbf{x}} = A_c \mathbf{x}, \quad (12)$$

using an ODE solver such as `ode45`. Because the solver uses $\Delta t_{\text{solver}} \ll \Delta t_{\text{DNS}}$, high-frequency components in the data are smoothed out.

2) *Analytic Semi-discrete-Time Solution*: The linear ODE admits a closed-form solution:

$$\mathbf{x}(t) = \Phi_{\text{DMD}} (\mathbf{b} \odot e^{\omega t}), \quad (13)$$

where $\mathbf{b} = \Phi_{\text{DMD}}^{-1} \mathbf{x}_1$. This method evaluates the exact continuous solution and may be sampled at arbitrary times, therefore producing discrete results that exactly give the predictions fit for the given data. Therefore, this prediction could be called semi-discrete.

3) *Discrete Iteration*: Two variants exist:

1) Using the inferred discrete operator:

$$\mathbf{x}_{k+1} = A_d \mathbf{x}_k. \quad (14)$$

2) Discretizing the continuous operator:

$$A_{\Delta t} = e^{A_c \Delta t}, \quad \mathbf{x}_{k+1} = A_{\Delta t} \mathbf{x}_k. \quad (15)$$

In this project, the operators were inferred using the **discrete-time formulation** but predictions were generated using the **semi-discretized operator**

$$\mathbf{x}(t) = \Phi_{\text{DMD}} (\mathbf{b} \odot e^{\omega t}), \quad (16)$$

Thus, a discrete-time estimation is paired with semi-discrete-time prediction.

C. DMD Performance

DMD reconstructs the training region with reasonable accuracy but exhibits limited predictive capability outside it due to coarse temporal resolution, and the limitations of linear modeling for turbulent flows.

IV. NON-LINEAR OPERATOR INFERENCE

The reduced-order dynamics of the POD/DMD coefficients $\mathbf{a}(t) \in \mathbb{R}^r$ are modeled using a quadratic ODE:

$$\dot{\mathbf{a}} = \mathbf{A}\mathbf{a} + \mathbf{H}(\mathbf{a} \otimes \mathbf{a}), \quad (17)$$

where \mathbf{A} (linear) and \mathbf{H} (quadratic) are inferred from data using regularized least squares. Two formulations exist for learning the operators, and several stability-enhancing corrections may be applied to improve prediction quality.

A. Discrete vs. Continuous-Time Operator Inference

Operator Inference may be performed in either discrete or continuous time. These formulations differ only in how the operators are inferred—the prediction step (ODE integration or discrete iteration) is separate.

1) *Discrete-Time OpInf*: Given snapshot pairs $(\mathbf{a}_k, \mathbf{a}_{k+1})$ with timestep Δt , the discrete reduced model is

$$\mathbf{a}_{k+1} = \mathbf{M}\mathbf{a}_k + \mathbf{Q}(\mathbf{a}_k \otimes \mathbf{a}_k), \quad (18)$$

where \mathbf{M} and \mathbf{Q} are identified using least squares. This formulation is numerically robust because the regression directly targets observable differences.

2) *Continuous-Time OpInf*: Continuous-time OpInf fits the ODE

$$\dot{\mathbf{a}}(t_k) = \mathbf{A}\mathbf{a}_k + \mathbf{H}(\mathbf{a}_k \otimes \mathbf{a}_k), \quad (19)$$

where $\dot{\mathbf{a}}(t_k)$ is estimated by finite differences. Although this preserves the continuous-time structure of the ROM, it is strongly sensitive to:

- discretization error in finite-difference derivatives,
- instability during long-time integration,
- decay of modal energy.

For turbulent flows, these issues lead to excessive damping in long-time predictions.

3) *Prediction Strategies: Three Continuous/Discrete Modes*: Once operators \mathbf{A} and \mathbf{H} are inferred prediction may be carried out using one of three methods in our project:

a) *ODE Integration (Fully Continuous Prediction)*:

Solve the CT-ROM using ODE45 or any stiffness-aware integrator:

$$\dot{\mathbf{a}} = \mathbf{A}\mathbf{a} + \mathbf{H}(\mathbf{a} \otimes \mathbf{a}). \quad (20)$$

Energy scaling may optionally be applied at each time step.

b) *Energy-Scaled Residual Corrected OpInf (ES-rCT-OpInf)*: Our proposed hybrid method combines the numerical robustness with the physical aspect of energy scaling:

$$\mathbf{a}_{k+1} = \mathbf{a}_k + \Delta t \left(\mathbf{A}\mathbf{a}_k + \mathbf{H}(\mathbf{a}_k \otimes \mathbf{a}_k) \right), \quad (21)$$

$$\mathbf{a}^*(t_k) = \alpha_k \mathbf{a}_{\text{pred}}(t_k), \quad (22)$$

$$\alpha_k = \sqrt{\frac{\|\mathbf{a}(t_k)\|_2^2}{\|\mathbf{a}_{\text{pred}}(t_k)\|_2^2}}. \quad (23)$$

The first equation is obtained by using `ode45` integrator, which integrates between two timesteps, and then the scaling is applied to the predicted values of the model, and then the scaled output is passed onto the next integration, which goes on like before. So, the only quantity we need to extract from the flow is the energy of the vector being studied.

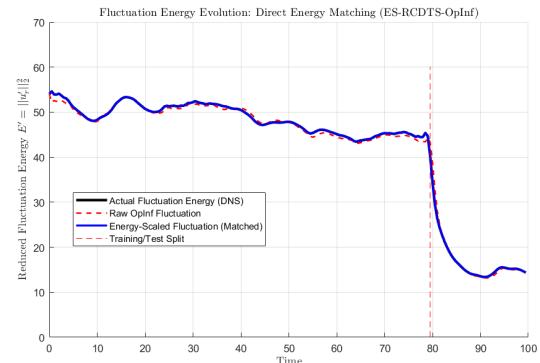


Fig. 3: Energy Matching through scaling

c) *Analytical Integration (Semi-Discrete Prediction)*:

Since we can't get a closed form solution of the non-linear model, it is not possible to solve this equation by this method.

d) *Discrete-Time Stepping (Fully Discrete Prediction)*:

A discrete-time ROM is constructed by Euler discretization:

$$\mathbf{a}_{k+1} = \mathbf{a}_k + \Delta t \left[\mathbf{A}\mathbf{a}_k + \mathbf{H}(\mathbf{a}_k \otimes \mathbf{a}_k) \right]. \quad (24)$$

Energy scaling may again be applied after each step. In our project, Continuous-Time Op-inf was used and prediction was done using different methods. In the first method, direct `ode45` was used to numerically integrate using adaptive timesteps. Along with this another prediction method of Energy-Scaled Residual Corrected Continuous time Operator Inference was used to predict the flow. Later an Euler discretization was used to discretely integrate and predict.

B. Effect of Large Sampling Timestep

The JHTDB data were extracted using $\Delta t = 0.25$, (and from autocorrelation plots, we have taken our timestep to be $\Delta t = 0.5$ larger than database) which is large compared to near-wall turbulence turnover times. This leads to:

- temporal aliasing of vortical structures,
- attenuation of important fluctuations,
- loss of high-frequency spectra.

Consequently, continuous-time Operator Inference with `ode45` predictions struggle to resolve the correct temporal scales of the turbulent flow when trained on such coarsely sampled data. This prompted me to explore other methods like ES-RCTS Opinf and Discrete time prediction which performed better in capturing the turbulent structures compared to the integrator.

C. Comparison with ODE Integrators

Although continuous-time OpInf defines a nonlinear reduced-order model

$$\dot{\mathbf{a}} = A \mathbf{a} + H(\mathbf{a} \otimes \mathbf{a}),$$

integrating this ODE using solvers such as `ode45` introduces artificial smoothing. Because the solver evaluates the ROM at time intervals much smaller than the DNS sampling interval, sub- Δt dynamics—which are not present in the training data—are implicitly filtered out. As a result, the reconstructed trajectories appear smoother than the true turbulent evolution.

V. SINDY-BASED DISCOVERY OF A REDUCED RANS X-MOMENTUM MODEL

To extract a compact, data-driven evolution model for the mean X-velocity $U(y, t)$, we apply Sparse Identification of Nonlinear Dynamics to a short DNS time series. Only $N_t = 5$ snapshots with spacing $\Delta t = 0.3$ are used, and all fields are projected onto a reduced grid (400 points) for numerical stability.

A. Library Construction and Targets

We build a physically consistent SINDy library containing the dominant terms of the RANS X-momentum equation:

$$\Theta = [-UU_x \quad -VU_y \quad -\frac{1}{\rho}P_x \quad \nu U_{yy} \quad \partial_y \langle u'v' \rangle]. \quad (25)$$

The Reynolds shear stress is estimated from the five instantaneous DNS fields,

$$\langle u'v' \rangle(y) = \frac{1}{N_t} \sum_{k=1}^{N_t} (u(\cdot, t_k) - \bar{U})(v(\cdot, t_k) - \bar{V}).$$

The regression target is the time derivative of U :

$$\frac{\partial U}{\partial t}(t_k) \approx \frac{U(t_{k+1}) - U(t_{k-1})}{2\Delta t}, \quad k = 2, \dots, N_t - 1. \quad (26)$$

All spatial derivatives (U_x, U_y, U_{yy}, P_x) are obtained using centered finite differences on the reduced grid.

B. Sparse Regression and Identified Model

The full regression matrix is assembled by stacking all spatial points and interior time indices. We apply standard sequential-thresholded least squares with parameters

$$\lambda = 10^{-3}, \quad i_{\max} = 15.$$

SINDy gives the following time-evolution model:

$$\begin{aligned} \frac{\partial U}{\partial t} = & 1.1996(-UU_x) + 0.3675(-VU_y) + 0.3597(-P_x/\rho) \\ & + 0.7878(\nu U_{yy}) + 1.0778(\partial_y \langle u'v' \rangle). \end{aligned} \quad (27)$$

VI. RESULTS AND DISCUSSION

A. Performance of ROM Methods

Figure 4 and Figure 5 compares the relative L2 errors of DMD, OpInf, and ES-RCDTS-OpInf and Discrete-Opinf. DMD has low error in the training as it just fits the observable linearly, but its prediction error is worse because its linear formulation cannot capture the nonlinear transitional dynamics, and it is done using a discrete formulation. Standard OpInf through `ode45` does a bad job in training and prediction due to attenuation of turbulent fluctuations, but there isn't much difference in training and prediction errors due to the mean flow being predicted the majority of the time, which remains constant. The ES-RCDTS-OpInf method has training error higher than discrete methods, but it does predict the best among all the models, as it is a continuous-time model used for prediction, and it is reinforced with energy at each time instant. The discrete-time integrating of Non-linear operator inference in Figure 5 does the best among all in training due to the capability of capturing non-linearities, and due to the discrete-time prediction, it does the best in training and is the second best in the test region, thereby saying that ES-RCTS Opinf is the most general model which could be used by only knowing the energy of the states. But still, this is not a sufficient model to analyse the coherent structures as they diminish with time. Hence, a model of higher fidelity is needed to model the Turbulent boundary layer.

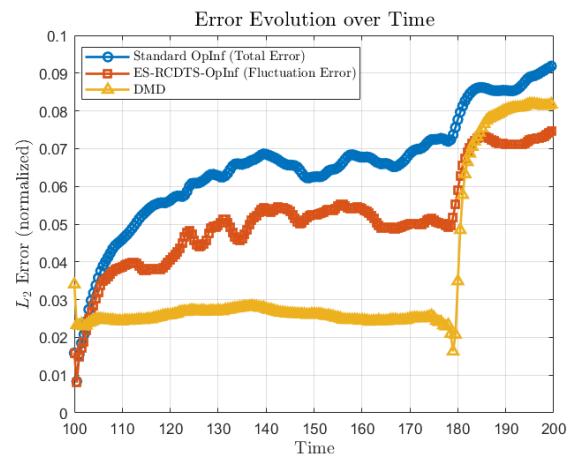


Fig. 4: ROM Training Error Comparison

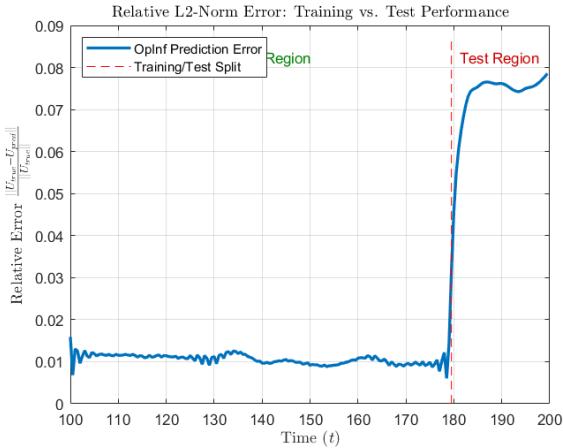


Fig. 5: Discrete time OpInf error

B. Skin-Friction Prediction

Figure 6 compares predicted skin-friction coefficients with classical laminar and turbulent [1],

$$C_f^{lam} = \frac{0.664}{\sqrt{Re_x}}, \quad C_f^{turb} = \frac{0.445}{(\log(0.06Re_x))^2}.$$

All ROMs capture the transitional evolution of skin friction, indicating that any model can be used to capture the near-wall behaviour accurately

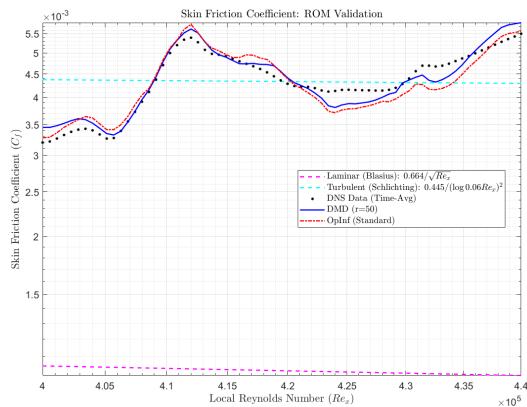


Fig. 6: Skin friction comparison.

C. SINDy Interpretation

- The discovered structure matches the physical RANS X-momentum balance, confirming that even very short DNS sequences contain enough information to recover dominant transport and turbulence-production mechanisms.
- The largest uncertainty arises from the Reynolds-stress gradient, since only five instantaneous snapshots are available for its estimation.
- As we increase the size of the library the prediction will go wrong because the uncertainty in the Reynolds

stress terms carries over to different terms, thereby giving spurious coefficients and false results.

- Therefore SINDy is not recommended for Boundary layer turbulence.

D. Computational Performance

All ROMs provide computational speedups relative to DNS. DMD is the fastest but least predictive. Standard OpInf and discrete OpInf requires with improved fidelity. ES-RCDTS-OpInf incurs a slightly higher cost but delivers the highest accuracy and most stable long-term predictions.

VII. CONCLUSION

The comparative analysis of reduced-order modelling techniques for transitional boundary-layer flows shows that, although DMD offers excellent training accuracy and high computational efficiency, its linear structure limits its predictive capability for nonlinear transitional dynamics. Standard OpInf shows attenuation of turbulent fluctuations requiring some changes in the model inference or prediction. The discrete nonlinear OpInf formulation performs strongly in training due to its ability to capture nonlinear terms directly in time. However, the ES-RCDTS-OpInf model provides the most robust and physically consistent predictions, achieving stable long-term behaviour through continuous-time energy reinforcement. Despite these advantages, all ROMs exhibit difficulty in maintaining coherent turbulent structures over extended times, indicating the need for more advanced, higher-fidelity reduced-order models for accurate turbulence capture.

VIII. ACKNOWLEDGEMENT

I acknowledge the use of ChatGPT [5] and Gemini AI [6] to reframe the sentences and rectify some of the codes.

REFERENCES

- [1] Y. Li, E. Perlman, M. Wan, Y. Yang, C. Meneveau, R. Burns, S. Chen, A. Szalay, and G. Eyink, “A public turbulence database cluster,” in *Proceedings of the ACM/IEEE Supercomputing Conference (SC08)*. Austin, TX: ACM/IEEE, 2008.
- [2] Ideal Simulations, “Turbulence models in cfd – rans, des, les, and dns,” <https://www.ideal simulations.com/resources/turbulence-models-in-cfd/>, accessed: 2025-11-25.
- [3] S. B. Pope, *Turbulent Flows*. Cambridge, U.K.: Cambridge University Press, 2000.
- [4] S. L. Brunton and J. N. Kutz, *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*. Cambridge, U.K.: Cambridge University Press, 2019.
- [5] OpenAI, “Chatgpt,” Large language model, 2025, available: <https://www.openai.com/chatgpt>.
- [6] Google, “Gemini,” Multimodal large language model, 2025, available: <https://ai.google.dev/gemini>.