XLVI CILAMCE
VITÓRIA - ES - BRAZIL - 2025
XLVI IBERO-LATIN AMERICAN CONGRESS ON COMPUTIONAL METHODS IN ENGENEERING

Online Model Order Reduction for Flow Simulations problems using Dynamic Mode Decomposition

Felipe I. M. Toledo [1], Gabriel F. Barros[1], Renato N. Elias[1], Alvaro L. G. A. Coutinho [1]

[1]*High Performance Computing Center and Civil Engineering, COPPE/Federal University of Rio de Janeiro*
Av. Athos da Silveira Ramos, 149 CT, Bloco B - 101, 21941-909, Rio de Janeiro, Brazil
{felipe.toledo, gabriel.barros, alvaro, rnelias}@coc.ufrj.br

Abstract

In this study, a reduced-order model (ROM) was directly coupled with a physical solver to predict system evolution and progressively replace the full-order model (FOM) solutions. Dynamic Mode Decomposition (DMD), implemented via the PyDMD library, was employed to perform the ROM predictions. The flow around a 2D cylinder with Reynolds number $Re = 100$ was chosen as the case study. The FOM solver uses the FEniCSx platform. The quality of the online DMD model approximation was evaluated using the residual norm generated exclusively by the DMD prediction, ensuring that velocity and pressure predictions remained within an acceptable tolerance. Additionally, the Frobenius norm was employed to monitor the accuracy of the velocity and pressure matrix predictions, assessing DMD's non-intrusive capability to capture and reproduce the system's dynamics. Experiments were conducted varying the number of dynamic modes required to achieve the desired accuracy. Using DMD as an alterative to the nonlinear solver reduced the computational time for the numerical simulations up to 20% with little to no loss in the solution quality.

Keywords: model order reduction, dynamic mode decomposition, scientific machine learning, computational fluid dynamics.

1 Introduction

Solving systems of partial differential equations with multiple time scales involves a high consumption of computational resources. For this reason, the search for methods that allow efficient solutions in terms of computational cost is crucial in numerical modeling. Traditionally, these problems have been approached using full-order models (FOM), in which numerical solvers are employed to calculate the variables that describe the system's physics. However, alternative approaches based on machine learning have emerged [1], which aim to approximate the solution by training models capable of learning physical patterns and predicting unseen scenarios.

Typically, machine learning models are divided into a training phase (usually referred to as the offline phase) and an inference phase (also known as the online phase). In the Reduced Order Modeling (ROM) community [2], the offline phase typically involves manipulating large datasets, while the online phase encompasses faster and computationally less intensive operations. This phase requires prior simulations using the full-order model (FOM), which corresponds to the physical solver, in order to gather the necessary data. This process entails high computational costs, limiting the scalability of the approach [3]. In contrast, integrating machine learning models into on-the-fly schemes - i.e., those that run simultaneously with the simulation, making projections as new data becomes available - has emerged as a promising alternative to replace the solutions provided by full-order physical simulators gradually [4]. This strategy reduces training costs, avoids the generation of unnecessary data, and optimizes the temporal advancement of the simulation.

Among the various reduced-order modeling (ROM) approaches, this study focuses on the application of the Dynamic Mode Decomposition (DMD) [5] algorithm, a non-intrusive factorization and dimensionality reduction technique specifically designed to extract spatiotemporal patterns in data. In its most common form, DMD processes high-dimensional sequential measurements to extract coherent structures, identify dynamic behavior, and reduce complex evolutionary processes to their dominant and essential components [6]. This technique has proven effective in a wide range of applications, particularly in the field of fluid mechanics. In this work, we consider DMD coupled with a numerical solver to perform on-the-fly predictions during simulation runtime following the ideas presented in [7]. This approach enables time-step skipping on the classical numerical solvers, producing approximations within a predefined level of accuracy, thus significantly reducing computational cost without compromising the quality of the results. We test our approach on the classical flow around a cylinder benchmark with $Re = 100$. This paper is structured as follows: Section 2 covers the governing equations and implementation details for the numerical simulations and test case. Information regarding DMD is also covered in this section. We show our results and discussion in Section 3. Finally, we summarize our findings in Section 4.

2 Methodology

In this section, we describe the problem formulation, the process of generating data for training and testing the model, and the details on how the ROM is constructed.

2.1 Governing equations

The viscous flow in the domain is described by the incompressible Navier-Stokes equations equipped with the prescribed Dirichlet and Neumann boundary conditions.

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} = -\nabla p + \nu \nabla^2 \mathbf{u} \quad \text{in } \Omega \times (0, T], \tag{1}$$

$$\nabla \cdot \mathbf{u} = 0 \quad \text{in } \Omega \times (0, T], \tag{2}$$

$$\mathbf{u} = \mathbf{u}_D \quad \text{in } \Gamma_D \times (0, T], \tag{3}$$

$$(-p\mathbf{I} + \nu \nabla \mathbf{u}) \cdot \mathbf{n} = \mathbf{t}_N \quad \text{in } \Gamma_N \times (0, T] \tag{4}$$

where t represents time, $\mathbf{u}$ is the velocity field, p is the pressure, ρ is the fluid density, and ν is the kinematic viscosity. The governing equations need an initial condition $\mathbf{u}(\mathbf{x}, 0) = \mathbf{u}_0(\mathbf{x})$. The flow simulation over a two-dimensional cylinder with $Re = 100$ is selected as a benchmark test case. The Navier-Stokes equation are approximated via the Finite Element Method using the FEniCSx package, an open-source finite element library with a high-level Python interface that efficiently solves partial differential equations [8]. To circumvent the LBB condition, we use a mixed finite element formulation, that is, P2-P1 Taylor-Hood elements.

2.2 Description of the DMD procedure

The temporal evolution of a variable $\mathbf{y} \in \Omega$, which fully describes the state of a fluid system, is governed by the nonlinear dynamical system $\mathbf{y}_{t+1} = \mathbf{F}(\mathbf{y}_t)$, where $\Omega \subseteq \mathbb{R}^d$ is the state space and $F : \Omega \to \Omega$ governs the system's evolution. DMD begins by collecting data in the form of snapshots and organizing them into matrices. Each snapshot is arranged as a column vector $\mathbf{y}_k$, and the matrices $\mathbf{Y}_1$ and $\mathbf{Y}_2$ represent the sequence of snapshots over time, defined as $\mathbf{Y}_1 = [\mathbf{y}_0, \ldots, \mathbf{y}_{n-1}] \in \mathbb{R}^{N \times n}$ and $\mathbf{Y}_2 = [\mathbf{y}_1, \ldots, \mathbf{y}_n] \in \mathbb{R}^{N \times n}$. We consider constructing a linear and approximate dynamical system seeking a matrix $\mathbf{A} \in \mathbb{C}^{d \times d}$ such that $\mathbf{Y}_2 \approx \mathbf{A}\mathbf{Y}_1$. The matrix $\mathbf{A}$ is found by solving the minimization problem $\min_{\mathbf{A} \in \mathbb{C}^{d \times d}} \|\mathbf{Y}_2 - \mathbf{A}\mathbf{Y}_1\|_F$, where $\| \cdot \|_F$ denotes the Frobenius norm. The solution is given by $\mathbf{A} = \mathbf{Y}_2\mathbf{Y}_1^\dagger$, with † the Moore–Penrose pseudoinverse.

The direct computation of the matrix $\mathbf{A}$ is often computationally expensive due to its high dimensionality; therefore, to reduce this complexity, the singular value decomposition (SVD) of the snapshot matrix $\mathbf{Y}_1$ is used, expressed as $\mathbf{Y}_1 = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^*$, where $*$ denotes the conjugate transpose, $\mathbf{U}$ contains the left singular vectors, $\mathbf{\Sigma}$ is the diagonal matrix of singular values, and $\mathbf{V}^*$ contains the right singular vectors. To achieve a low-rank approximation, only the first r singular values and corresponding vectors are considered, yielding the approximation $\mathbf{Y}_1 \approx \mathbf{U}_r\mathbf{\Sigma}_r\mathbf{V}_r^*$, where $\mathbf{U}_r \in \mathbb{R}^{N \times r}$, $\mathbf{\Sigma}_r \in \mathbb{R}^{r \times r}$, and $\mathbf{V}_r \in \mathbb{R}^{M \times r}$. The choice of this reduced-rank approximation has been the subject of extensive research due to its impact on the efficiency and accuracy of the method [9]. Once the low-rank decomposition of the matrix $\mathbf{Y}_1$ is obtained, the reduced matrix can be computed as $\tilde{\mathbf{A}} = \mathbf{U}_r^*\mathbf{Y}_2\mathbf{V}_r\mathbf{\Sigma}_r^{-1}$, where $\mathbf{U}_r$, $\mathbf{\Sigma}_r$, and $\mathbf{V}_r$ are derived from the truncated SVD of $\mathbf{Y}_1$. This reduced matrix preserves the main dynamical features of the system with significantly lower computational cost. The spectral decomposition of $\tilde{\mathbf{A}}$ is then performed as $\tilde{\mathbf{A}}\mathbf{W} = \mathbf{W}\mathbf{\Lambda}$, where $\mathbf{\Lambda}$ contains the eigenvalues and $\mathbf{W}$ the eigenvectors of $\tilde{\mathbf{A}}$. The dynamic modes are computed as $\mathbf{\Phi} = \mathbf{Y}_2\mathbf{V}_r\mathbf{\Sigma}^{-1}\mathbf{W}$, which represent spatial-temporal coherent patterns of the system. The system's temporal evolution can be reconstructed or predicted using $\tilde{\mathbf{Y}}_1(k\Delta t) = \mathbf{\Phi}\mathbf{\Lambda}^k\mathbf{b}_0$, where $\mathbf{b}_0$ is the initial condition projected onto the modal space, Δt is the time step, and k is the discrete time index.

2.3 Workflow

Figure 1 presents the workflow used to incorporate a linear approximation obtained through our reduced order model (ROM), here, DMD, together with the full-order model (FOM). Analogous to a transient problem, the simulation begins with time integration from an initial condition. In the first time step, only one snapshot of pressure and velocity is available; therefore, the system cannot yet be approximated using the ROM, and it is necessary to solve it using the FOM. Since this is a coupled approach, the pressure and velocity matrices are stored separately in order to avoid numerical consistency issues.

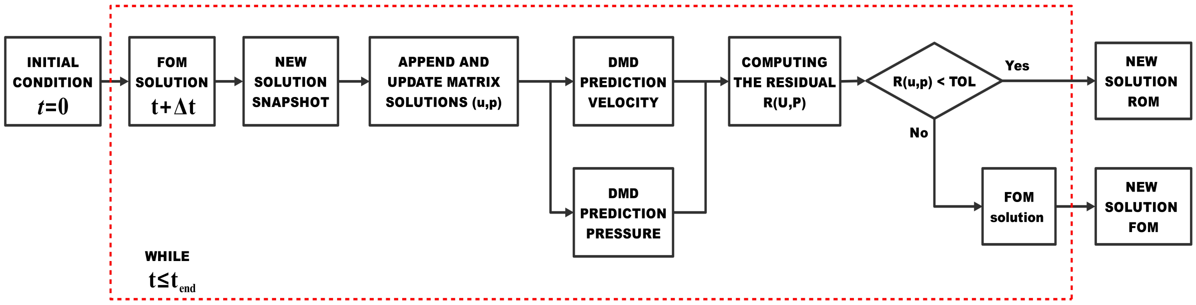


Figure 1. Flow diagram for online DMD coupled with numerical simulator

From the second time step onward, the reduced-order model (ROM) is initialized as a predictor for both pressure and velocity, using the corresponding *snapshot* matrices. Then, a one-step-ahead prediction is made, and the L2 norm of the residual $R(\mathbf{u}, p)$ is evaluated in the global equation. If this norm exceeds the established tolerance, the prediction is not accepted, and the full-order solver (FOM) is called again to update the matrices with the obtained solution. On the other hand, if the norm of the residual is below the tolerance, the prediction is considered valid, and it replaces the FOM, which is then used to update the coupled solution vector for both pressure and velocity, thereby starting a new simulation cycle. A common way to assess the accuracy of a prediction is through the local error, defined at time step k as $\mathbf{e}_k = \mathbf{y}_k - \hat{\mathbf{y}}_k$, where the numerical vector $\mathbf{y}_k = [\mathbf{u}_k \quad \mathbf{p}_k]$ denotes the coupled vector combining the velocity field $\mathbf{u}_k$ and the pressure field $\mathbf{p}_k$ and $\hat{\mathbf{y}}_k$ represents the DMD approximation of $\mathbf{y}_k$. However, this evaluation requires access to both the reference solution obtained by the full-order model (FOM) and the approximate solution from DMD, which can be computationally expensive. As an alternative, the residual is proposed as a representative measure of the local error, enabling the assessment of DMD prediction quality without requiring the exact solution at each time step. Let $\mathcal{L}(\mathbf{u}, p) = \mathbf{f}$ be a system governed by a differential operator. For an approximate solution $(\hat{\mathbf{u}}_k, \hat{p}_k)$, the residual is defined as $\mathbf{R}(\hat{\mathbf{u}}_k, \hat{p}_k) = \mathcal{L}(\hat{\mathbf{u}}_k, \hat{p}_k) - \mathbf{f}$. The differential operator corresponds to the discretized version of the Navier–Stokes equations. The residual norm quantifies the magnitude of the local errors' contribution, thus serving as a metric for the approximation. In this work, the residual is computed using the DMD-prediction velocity and pressure fields at the future time step, together with the terms from the previous FOM step, allowing for a single-step prediction for the transition between the reduced-order model (ROM) prediction and the high-fidelity physical solver (FOM).

2.4 Description of the experiment

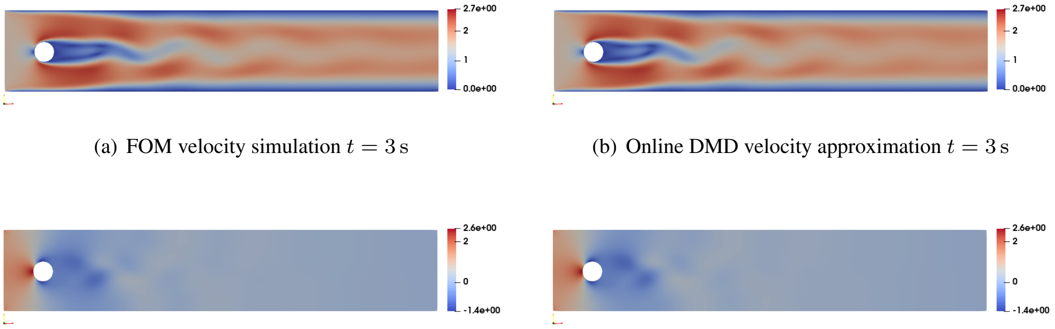
The flow is described in a Cartesian coordinate system where the x-axis is aligned with the streamwise direction and the y-axis is transverse and orthogonal to the cylinder axis, in a two-dimensional domain representing a rectangular channel with length $l = 2.2$ and height 0.41. The boundary conditions impose a uniform inflow velocity $\mathbf{u} = (1.5, 0)$ at the inlet boundary Γ_{in} (located at $x = 0$), corresponding to a Reynolds number $Re = 100$. No-slip conditions, meaning zero velocity, are applied on the channel walls Γ_{walls} (at $y = 0$ and $y = H$) and on the cylinder surface Γ_{obs}. Finally, a zero pressure condition $p = 0$ is prescribed at the outlet boundary Γ_{out}. The cylinder has a radius $r = 0.05$ and is located at position $(c_x, c_y) = (0.2, 0.2)$. An unstructured mesh was used to refine elements to a minimum size $h_{\min} = \frac{r}{2}$ in regions close to the cylinder, i.e., for distances $d \leq r$. Outside this zone, the element size gradually increases up to a maximum $h_{\max} = 0.1H$ for distances $d \geq H$. The simulation was conducted on a two-dimensional mesh comprising 3968 cells and 4132 vertices. A mixed finite element space

$W = V \times Q$ was used, where V is a vector-valued Lagrange space of degree 2 for velocity and Q is a scalar Lagrange space of degree 1 for pressure. The resulting system has 16,200 degrees of freedom for velocity and 4,132 for pressure. Time integration is performed using a Backward Euler scheme with a fixed time step of 0.01 [s] and a time interval from 0 to 3 [s]. The problem is solved monolithically, that is, velocity and pressure are coupled, using the nonlinear Newton method available in DOLFINx. At each Newton iteration, the resulting linear system is solved using the GMRES iterative solver, preconditioned by an incomplete LU factorization. The nonlinear solver convergence criteria are defined by a relative residual tolerance of 10^{-6}, an absolute residual tolerance of 10^{-6}, and a maximum of 50 iterations. Here, the convergence is monitored using the relative L^2-norm of the nonlinear residual vector. Simultaneously with the configuration of the physical solver, the `PyDMD`[1] library [10] was employed to perform Dynamic Mode Decomposition (DMD) and predict the velocity and pressure vectors into future time steps. The DMD model is initialized once a temporal window of 30 time steps is available. This value is arbitrarily chosen to ensure an acceptable number of snapshot columns in the system. From this point onward, the model begins operating online, providing predictions of the velocity and pressure fields, with the residual norm remaining close to the tolerance value defined as 10^{-3}. To assess the influence of the low-rank approximation, we evaluated the impact of varying the parameter r, which determines the rank of the approximation of the operator $\tilde{\mathbf{A}}$. Four values were tested: 40, 60, 80, and an optimal value. This optimal value, described as $r =$ opt*, corresponds to a variable rank determined by the data-driven criterion proposed in [11]. In this case, the rank adapts based on incoming snapshots using predefined heuristics [11] built in the PyDMD library to optimize the choice of r. That is, for a variable time window (which is the case of an ongoing numerical simulation), every time the DMD routine is invoked, an optimal value of r is chosen for each prediction. After a fixed time interval, two metrics were used to evaluate the model's performance: (i) the number of time steps for which the DMD model remained within the predefined residual norm tolerance (10^{-3}), and (ii) the total reconstruction error, quantified as the relative difference between the FOM-DMD strategy and solely the FOM solution vectors. These evaluations enable a systematic analysis of the trade-off between computational efficiency and fidelity when integrating DMD-based projections into real-time numerical simulations. The relative error ε_p in the Frobenius norm between the original data matrix $\mathbf{Y}$ and the FOM-DMD approximations matrix $\hat{\mathbf{Y}}$ over the entire simulation is given by:

$$\varepsilon_p(\hat{\mathbf{Y}}) = \frac{\|\mathbf{Y} - \hat{\mathbf{Y}}\|_F}{\|\mathbf{Y}\|_F} \tag{5}$$

3 Results and Ablation Studies

In this section, we present and discuss the results obtained with the developed model. To this end, we begin with a visual comparison between the original flow and the approximate prediction of the velocity and pressure fields at the final time step, where a DMD model with optimal rank $r = 135$ and exact mode enabled was applied, as implemented in the PyDMD package. Figure 2 shows this comparison, where one can visually observe the similarity between both fields when comparing the FOM simulation to the results obtained via DMD.



(a) FOM velocity simulation $t = 3\,\mathrm{s}$

(b) Online DMD velocity approximation $t = 3\,\mathrm{s}$

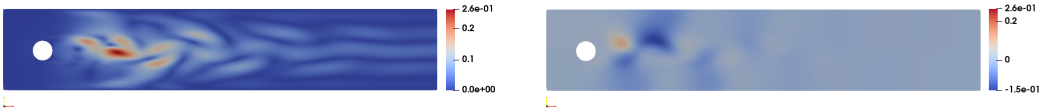
(c) FOM pressure simulation $t = 3\,\mathrm{s}$

(d) Online DMD pressure approximation $t = 3\,\mathrm{s}$

Figure 2. Comparison between the original FOM simulation and the online DMD approximation for both the velocity and pressure fields at $t = 3\,\mathrm{s}$

[1] https://github.com/PyDMD/PyDMD

Building upon the previous comparison, Figure 3 shows the local error between the velocity and pressure fields obtained using the Full-Order Model (FOM) and the FOM coupled with Dynamic Mode Decomposition (FOM–DMD). The discrepancies, mainly located in the vortex-generation region, reflect the complexity of the underlying dynamics. Although the error magnitudes are relatively low, pressure predictions show similar values but over smaller areas, indicating better predictions in the pressure field reconstruction.



(a) local error velocity $t = 3\,\mathrm{s}$

(b) local error pressure $t = 3\,\mathrm{s}$

Figure 3. Error in the velocity and pressure approximations computed at each node of the domain

To further assess the ROM performance, we test our strategy using different values of r. Table 1 consolidates the results for $r = \{40, 80, 120\}$ as well as the case where DMD optimizes r during the simulation runtime (Opt*). We assess how often on-the-fly predictions have residuals within the prescribed tolerance and the Frobenius relative errors between the Full-Order Model (FOM) solutions and those from the FOM-DMD approach. As r increases, the time-step prediction quality improves, indicating a more accurate local approximation by the ROM. Despite this, Frobenius errors for pressure and velocity remain nearly constant due to the low residual norm tolerance, ensuring reliable predictions. We noticed that for this optimal case, 135 time steps - about 45% of the total 300 time steps - were processed using DMD instead of requiring the computation of the FOM model. This suggests that dynamic rank models offer better local approximation and adaptive efficiency than static rank models, as they more effectively capture temporal system changes.

Table 1. Number of ROM predictions within the prescribed tolerance and Frobenius relative errors (in percentage) for pressure and velocity fields across different ranks using DMD

Rank	**Predictions with DMD**	**Frobenius relative error for velocity [%]**	**Frobenius relative error for pressure [%]**
40	38	1.5158	0.2245
60	55	1.5138	0.2245
80	90	1.5087	0.2451
Opt*	135	1.4081	0.2452

The optimal decomposition rank for both the velocity and pressure matrices varies at each time step during the simulation, reflecting the dynamic modes needed to best approximate the numerical solutions. Notably, both matrices exhibit linear behavior with respect to r and share the same size, highlighting a consistent pattern in their dynamic rank evolution. Figure 4 illustrates these variations throughout the simulation.

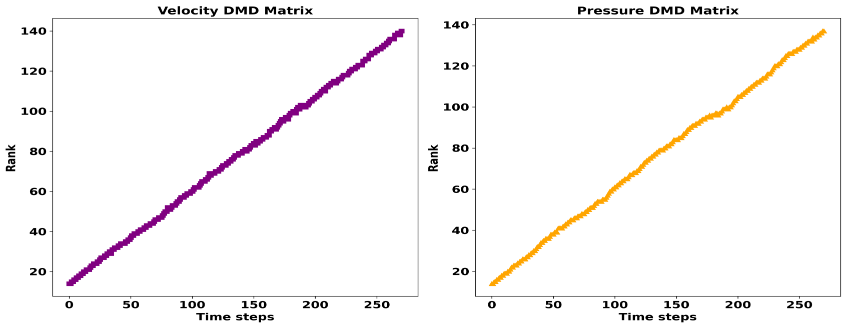
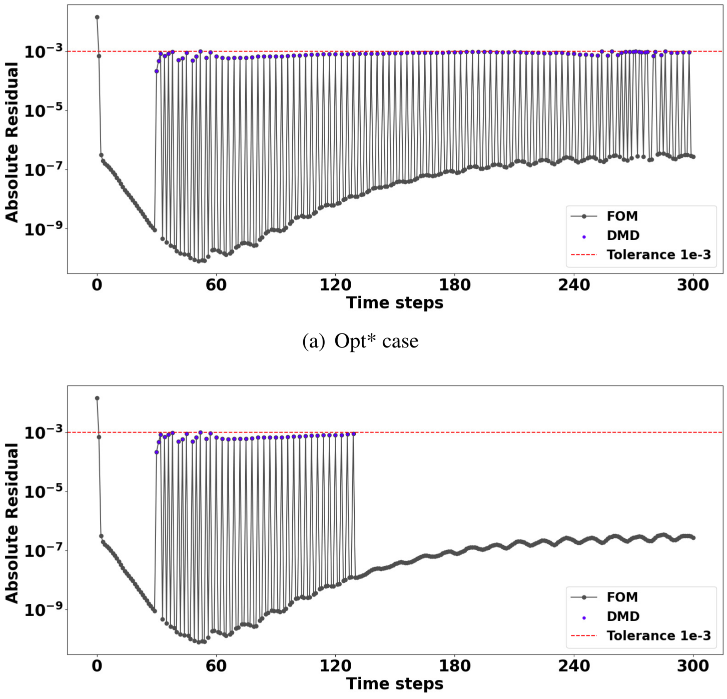


Figure 4. Time History of Optimal DMD Matrix Rank for Velocity (left) and Pressure (right)

Concerning the evolution of errors over time, Figure 5 shows the evolution of the residual norm, computed from the solution vector containing pressure and velocity, throughout the simulation. The red dashed line indicates the set tolerance level, while blue markers represent the approximations obtained via DMD and gray markers correspond to the Full-Order Model (FOM) solutions. The initial time interval marks the temporal window used to initialize the method. It can be observed that the DMD model successfully captures the underlying dynamics and provides predictions for velocity and pressure only within this initialization window. In contrast, the FOM delivers results over the entire simulation duration. In Figure 5(a), the dynamic adjustment of the rank allows the simulation to advance over more time steps, enabling a more accurate capture of the system's dynamics throughout the entire simulation. On the other hand, Figure 5(b), which uses a constant rank ($r = 40$), achieves accurate predictions only up to a certain point in time, after which the residual begins to grow and exceeds the predefined tolerance level. It is worth noting that although the case with an optimal rank (Opt*) allows the simulation to proceed further, it also exhibits a more pronounced increase in the residual at later stages compared to the fixed-rank case. This indicates a trade-off between maintaining accuracy and extending the time horizon when using adaptive reduction strategies.



(a) Opt* case

(b) $r = 40$

Figure 5. Time history of the residual norm for the coupled FOM-ROM

Finally, regarding efficiency, Table 2 presents the total execution time and speedup achieved for different DMD ranks. As the number of dynamic modes increases, the prediction becomes more accurate (as previously demonstrated), while also enhancing overall computational efficiency. Although adding more modes is known to increase the computational cost for the DMD algorithm, the increase in acceptable ROM solutions still reveals better efficiency in the system. The optimal configuration achieves the highest speedup with the lowest computational time among all cases.

Table 2. Total simulation time and speedup for DMD predictions using different ranks

Rank	ROM Time [s]	FOM Time [s]	Total Time [s]	Speedup
FOM	–	900	900	–
40	27	786	823	1.097
60	105	735	840	1.071
80	223	630	853	1.055
Opt*	250	510	760	1.184

4 Conclusions

This work demonstrates the effectiveness of Dynamic Mode Decomposition (DMD) in replacing the full-order model over several time steps, contributing to a reduction in the simulation time of the two-dimensional flow around a cylinder at a Reynolds number of 100. Using data from a full-order model (FOM), the DMD approach generates accurate predictions at each time step while maintaining physical consistency by controlling residual norms. This hybrid modeling strategy enables the progressive replacement of the traditional solver, resulting in a computational speedup of around 20% with little to no loss in accuracy. As the simulation progresses and the system becomes increasingly complex, we observed that using an adaptive number of modes yields better results compared to fixed values for r. Larger speedup values can be achieved if more effective on-the-fly strategies are employed for optimal performance (i.e., incremental SVD algorithms). These results highlight DMD as a promising real-time model order reduction technique for applications in fluid dynamics.

Acknowledgements. This study was financed in part by CAPES, Brazil - Finance Code 001. This work is also partially supported by FAPERJ, and CNPq.

Authorship statement. The authors hereby confirm that they are the sole liable persons responsible for the authorship of this work, and that all material that has been herein included as part of the present paper is either the property (and authorship) of the authors, or has the permission of the owners to be included here.

References

[1] S. L. Brunton, B. R. Noack, and P. Koumoutsakos. Machine learning for fluid mechanics. *Annual review of fluid mechanics*, vol. 52, n. 1, pp. 477–508, 2020.

[2] W. H. Schilders, Van der H. A. Vorst, and J. Rommes. *Model order reduction: theory, research aspects and applications*, volume 13. Springer, 2008.

[3] C. Meng, S. Griesemer, D. Cao, S. Seo, and Y. Liu. When physics meets machine learning: A survey of physics-informed machine learning. *Machine Learning for Computational Science and Engineering*, vol. 1, n. 1, pp. 20, 2025.

[4] D. Panchigar, K. Kar, S. Shukla, R. M. Mathew, U. Chadha, and S. K. Selvaraj. Machine learning-based cfd simulations: a review, models, open threats, and future tactics. *Neural Computing and Applications*, vol. 34, n. 24, pp. 21677–21700, 2022.

[5] J. H. Tu. *Dynamic mode decomposition: Theory and applications*. PhD thesis, Princeton University, 2013.

[6] Z. Wu, S. L. Brunton, and S. Revzen. Challenges in dynamic mode decomposition. *Journal of the Royal Society Interface*, vol. 18, n. 185, pp. 20210686, 2021.

[7] S. W. Suh, S. W. Chung, P.-T. Bremer, and Y. Choi. Accelerating flow simulations using online dynamic mode decomposition. *arXiv preprint arXiv:2311.18715*, 2023.

[8] I. A. Baratta, J. P. Dean, J. S. Dokken, M. Habera, J. HALE, C. N. Richardson, M. E. Rognes, M. W. Scroggs, N. Sime, and G. N. Wells. Dolfinx: the next generation fenics problem solving environment, 2023.

[9] M. J. Colbrook. The multiverse of dynamic mode decomposition algorithms. In *Handbook of numerical analysis*, volume 25, pp. 127–230. Elsevier, 2024.

[10] S. M. Ichinaga, F. Andreuzzi, N. Demo, M. Tezzele, K. Lapo, G. Rozza, S. L. Brunton, and J. N. Kutz. Pydmd: A python package for robust dynamic mode decomposition. *Journal of Machine Learning Research*, vol. 25, n. 417, pp. 1–9, 2024.

[11] M. Gavish and D. L. Donoho. The optimal hard threshold for singular values is $4/\sqrt{3}$. *IEEE Transactions on Information Theory*, vol. 60, n. 8, pp. 5040–5053, 2014.