

Dynamic mode decomposition for density-driven gravity current simulations

Gabriel F. Barros¹, Adriano M. A. Côrtes², Alvaro L. G. A. Coutinho¹

¹*Programa de Engenharia Civil, COPPE/Universidade Federal do Rio de Janeiro
Avenida Horácio Macedo, CT Bloco B, 2030, Rio de Janeiro, RJ 21941-450, Brasil*

gabriel.barros, alvaro@coc.ufrj.br

²*NUMPEX-COMP, Universidade Federal do Rio de Janeiro*

*Rodovia Washington Luis, km 104,5, 25265-970, Duque de Caxias, RJ 25265-970, Brasil
adrimacortes@gmail.com*

Abstract. In the present work, we evaluate the capability of the Dynamic Mode Decomposition method to extract spatio-temporal coherent structures of density-driven gravity current simulations. The coupled problem is solved using the finite element method, and DMD is applied in the concentration dataset instead of all the simulation data. We generate reduced order models from the DMD with a different number of computed basis vectors and evaluate their accuracy and performance to capture the dynamics from the original system. We also extend our analysis to two relevant quantities of interest, the front position and mass conservation, testing the accuracy of the surrogate models. We also evaluate three different implementations of the Singular Value Decomposition, the core procedure in the DMD, in terms of performance and accuracy. Furthermore, we investigate the extrapolation ability of the Dynamic Mode Decomposition and evaluate its accuracy when considering different initial conditions.

Keywords: Dynamic Mode Decomposition, Singular Value Decomposition, Density-driven gravity flow

1 Introduction

Surrogate models with reduced dimensionality became of great interest among scientists and engineers of many different fields with the increasing complexity of the problems of interest. In the fluid dynamics community, the quantitative investigation of complex signals, either numerical or experimental, obtained from nonlinear systems is a widely addressed topic [1]. Although fluid flows are governed by infinite-dimensional partial differential equations (the Navier-Stokes equations) and accurate numerical approximations for complex fluid flow simulations still lead to high-dimensional finite spaces, it is well known that the main, most energetic, features of the flow are embedded in low-dimensional manifolds [2, 3]. The identification of these low-order dynamics can be obtained through “modal decomposition” techniques such as the Proper Orthogonal Decomposition (POD) [4, 5], Balanced Proper Orthogonal Decomposition (BPOD)[6], the Reduced Basis (RB) Method [7] and the Dynamic Mode Decomposition (DMD) [8–11], where the underlying physics of the system can be represented as a superposition of proper basis vectors. Each of these methods has its advantages and disadvantages and mathematical foundations, where the generation of the basis can be obtained from different approaches and have different interpretations.

In the present work, we focus on using DMD on density-driven gravity flows generated by two fluids. Particularly we focus on computing the position of the front of the current and verifying mass conservation. We generate surrogate models - or reduced order models (ROMs) - with different numbers of modes that inherit the low-rank structures of the binary flow presented in the high dimensional data and evaluate their ability to compute our quantities of interest (QoIs) with reasonable accuracy. We observe that even with a small number of modes, we achieve accurate results with models generated purely from data with small computational effort. We also compare three different algorithms for the Singular Value Decomposition (SVD), the core operation in the DMD, in terms of accuracy and performance. Moreover we test the DMD prediction capability by using the computed basis to evaluate the evolution of the concentration field for a different initial concentration field. The remainder of this paper is organized as follows. Section 2 briefly reviews the DMD method. The numerical experiments are given in Section 3. The paper ends with a summary of our main findings.

2 Dynamic Mode Decomposition

Dynamic Mode Decomposition is an equation-free, data-driven method that provides accurate assessments of the spatio-temporal coherent structures in a given complex system or short-time future estimates of such a system. In the fluid dynamics context, DMD has been applied to a wide variety of flow geometries (jets, cavity flow, wakes, channel flow, boundary layers, etc.) to study different phenomena [10]. It is one of the many machine learning methods related to dimensionality reduction on fluid dynamics data [3] and, among its applications, the DMD has been used mainly for structure extraction from fluids data and control-oriented methods. However, the ability of the DMD to synthesize data from simulations or experiments into ROMs with good extrapolation properties is still an active research topic, although several works presented advances in this direction [12, 13].

The DMD consists on splitting the dataset $\mathbf{Y} = [\mathbf{y}(t_0) \dots \mathbf{y}(t_m)]$ ordered in time into two datasets $\mathbf{Y}_1 = [\mathbf{y}(t_0) \dots \mathbf{y}(t_{m-1})]$ and $\mathbf{Y}_2 = [\mathbf{y}(t_1) \dots \mathbf{y}(t_m)]$, where m is the total number of observations in time of the dynamical system, and obtaining a linear mapping (matrix) \mathbf{A} from dataset \mathbf{Y}_1 to dataset \mathbf{Y}_2 , that is, $\mathbf{Y}_2 = \mathbf{A}\mathbf{Y}_1$. The computation of \mathbf{A} can be done as $\mathbf{A} = \mathbf{Y}_2\mathbf{Y}_1^\dagger$, where \mathbf{Y}_1^\dagger is the Moore-Penrose pseudoinverse of \mathbf{Y}_1 . Since fluid dynamics problems are often high-dimensional, the computation of the full Moore-Penrose pseudoinverse $m \times n$ could be expensive as well as highly ill-conditioned. In this case, we could consider computing $\tilde{\mathbf{A}}$, a $r \times r$ projection of \mathbf{A} where $r < m \ll n$. This can be done by factorizing the \mathbf{Y}_1 into a SVD such that $\mathbf{Y}_1 = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ and truncate the generated matrices by rank r . We can then obtain $\tilde{\mathbf{A}}$ as,

$$\tilde{\mathbf{A}} = \mathbf{U}_r^T \mathbf{Y}_2 \mathbf{V}_r \mathbf{\Sigma}_r^{-1}. \quad (1)$$

With the computation of $\tilde{\mathbf{A}}$ the matrix $\mathbf{\Psi}$ containing the DMD modes (technically, the Koopman modes [10]) can be extracted from the dataset with the relation,

$$\mathbf{\Psi} = \mathbf{Y}_2 \mathbf{V}_r \mathbf{\Sigma}_r^{-1} \mathbf{W}, \quad (2)$$

where \mathbf{W} are the eigenvectors of $\tilde{\mathbf{A}}$. The signal reconstruction can be done as $\mathbf{y}(t) \approx \tilde{\mathbf{y}}(t) = \mathbf{b}\mathbf{\Psi} \exp(\mathbf{\Omega}t)$, being \mathbf{b} the vector containing the projected initial conditions such that $\mathbf{b} = \mathbf{\Psi}^\dagger \mathbf{y}(t_0)$, and $\mathbf{\Omega}$ is a diagonal matrix whose entries are the eigenvalues $\omega_i = \ln(\lambda_i)/\Delta t$, where λ_i is an eigenvalue of $\tilde{\mathbf{A}}$, and Δt the time step size.

DMD relies on a sequence of linear algebra operations and one of them - the SVD - is directly responsible for the dimensionality reduction of the problem. SVD can represent a significant part of the computational effort of the code, meaning that improving SVD performance leads to significant CPU time gains. There are various algorithms for this purpose. In the present work, we implement the randomized SVD (rSVD for short) algorithm [14], a non-deterministic algorithm able to compute the near-optimal low-rank approximation of a given large dataset with good efficiency. The implementation of the rSVD is the first step for future different applications possibilities such as the use of DMD for real-time computations considering low-rank updates on an existing SVD [15]. The rSVD presents two features: oversampling, adding p columns to the random projection matrix leads to an accuracy gain of the algorithm (as a rule of thumb, 5 to 10 extra columns suffice), and power iteration, to force the decay of singular values when needed, where the number of power iterations q is an input parameter (usually less than 5). Our implementation relies on built-in functions of NumPy [16], a high-performance Python package with C bindings, for the key computational costs such as matrix products.

3 Numerical experiments

In this section, we evaluate the DMD use on density-driven gravity flows. The simulation consists of a lock-exchange between two fluids, the heavy fluid, A, and the lighter fluid, B, based on the numerical example in [17]. The difference between their densities is such that the Boussinesq hypothesis is considered valid. Moreover, particles in the heavy fluid have negligible inertia and are much smaller than the smallest length scales of the buoyancy-induced fluid motion. Thus, the dimensionless governing equations are,

$$\begin{aligned} \nabla \cdot \mathbf{u} &= 0, \\ \frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} + \nabla p - \frac{1}{\sqrt{Gr}} \Delta \mathbf{u} - \phi \mathbf{e}^g &= 0, \\ \frac{\partial \phi}{\partial t} + \mathbf{u} \cdot \nabla \phi - \frac{1}{Sc\sqrt{Gr}} \Delta \phi &= 0, \end{aligned} \quad (3)$$

where \mathbf{u} is the fluid velocity, ϕ is the concentration field, p is the pressure, $\mathbf{e}^g = (0, -1)$ is the vector pointing in the direction of gravity, $Sc = 1.0$ is the Schmidt number and $Gr = 5 \times 10^6$ is the Grashof number, two dimensionless numbers that relate viscous effects with diffusion and buoyancy effects, respectively. A Grashof number of this magnitude indicates a turbulent flow. The field $\phi = \phi_A/\phi_B$ is the concentration and is responsible for mapping the evolution of fluid interactions. We consider a tank, that is, a rectangular domain with length $L_1 = 18\text{m}$, height $L_2 = 2\text{m}$. The initial conditions are such that the heavy fluid is represented as a column with dimensions $L_x^0 \times L_y^0 = 1\text{m} \times 2\text{m}$ located at the left border of the tank and the light fluid fills the rest of the domain.

To solve the governing equations, we consider a finite element method spatial discretization and the finite difference method to approximate the evolution of the fields in time. We use the FEniCS 2019.1 [18] framework to generate our full order model (FOM) data. To circumvent the LBB-condition, using equal-order interpolation functions for the velocity-pressure pair, and avoid spurious oscillations in advection-dominated flows, we consider the residual based variational multiscale formulation of the Navier-Stokes equation. The concentration equation is predominantly advective, and then we apply the SUPG stabilization to its original Galerkin formulation. Details of the formulation can be found in [19]. We consider a mesh with 700×100 cells, where each cell is divided into two linear triangles.

In this example, we study the use of the DMD for different numbers of Koopman modes, r . We consider the solution of the finite element approximated equations as our full order model (FOM) and aim to build reduced order models (ROMs) that can work as surrogate models satisfying mass conservation and reproducing the evolution of the front position. Furthermore, we investigate the influence of three algorithms, in terms of performance and accuracy, for the Singular Value Decomposition, the DMD core operation. Also, instead of considering the totality of the simulated data from our FOM - that is, velocity, pressure, and concentration - for the generation of our reduced basis, we only take into account the concentration field, reducing the amount of processed data in the DMD by 75%. The next step is to define the number of Koopman modes, r , that our ROMs will inherit. According to [20], a good *a priori* estimate for r is by considering the value of

$$\kappa = \frac{\sum_{i=1}^r \sigma_i}{\sum_{i=1}^m \sigma_i} \geq \tau, \quad (4)$$

the retained energy of the low-rank approximation, where m is the rank and σ_i is the i -th singular value of \mathbf{Y}_1 , and τ is a prescribed tolerance such as 0.999999. That is, as r approaches m , κ approaches 1.0, and the truncated matrix inherits more information regarding the original dataset. The singular values of the concentration dataset are shown in Fig. 1. The figure shows the decay of the singular values as r increases and the value of κ for different chosen values of r . It is important to reiterate that the use of this *a priori* estimates gives us a good starting point to select r , but it is not directly related to any *a posteriori* error estimation relative to the reconstructed solution.

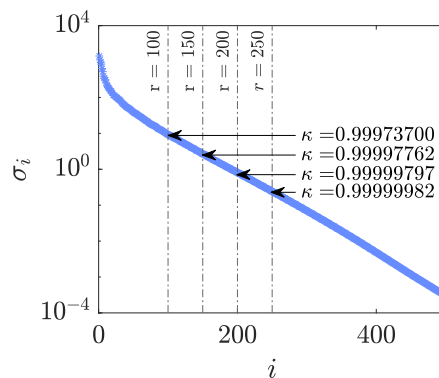


Figure 1. Singular values of the snapshots matrix, our selection of r and their respective values of κ .

We now reconstruct our solutions using DMD. As a first step, we compare the accuracy and performance of three SVD algorithms: NumPy SVD [16], Scikit-learn rSVD [21], and our implementation of the rSVD. NumPy SVD function invokes the high-performance LAPACK routine `_gesdd` while both rSVD algorithms are the implementation of the method proposed in [14] and are solved using 1 power iteration and 5 additional vectors for oversampling. Figure 2 shows the relative error and speed-up achieved by each algorithm for increasing values of r . In the figure, the NumPy SVD is identified simply by SVD, the Scikit-learn by SKSVD, and our implementation

by rSVD. The relative error (η) is computed as the Frobenius norm of the difference between the matrix whose columns are obtained from reconstructed solution and the original dataset \mathbf{Y} , divided by the Frobenius norm of the original dataset. The speed-up is evaluated as the ratio between the DMD CPU time and the FOM CPU time (including I/O), for increasing r . We note in Figure 2(a) that the three SVD algorithms presented the same behavior with negligible differences, revealing an exponential decay of the error while increasing the truncation rank r . We can also observe that the original implementation of the SVD is slower than the rSVD implementations for increasing r . The rSVD presented similar results, meaning that - for this given dataset size - our implementation of the rSVD is competitive when compared with the existent in a high-performance library in Python. From Figure 2(b), we observe that the DMD implementation using Numpy SVD is around 70 times faster than the FOM while the use of the rSVD increases this factor to 150 – 275. We notice, however, that the rSVD implementations present a significant speed-up decrease with increasing r in comparison with the SVD implementation, in a sense that a DMD with $r = 100$ considering our implementation of the rSVD is twice as faster than the same code running with $r = 250$. Therefore, the choice of the rank r is important for the efficiency of the DMD when considering the randomized SVD.

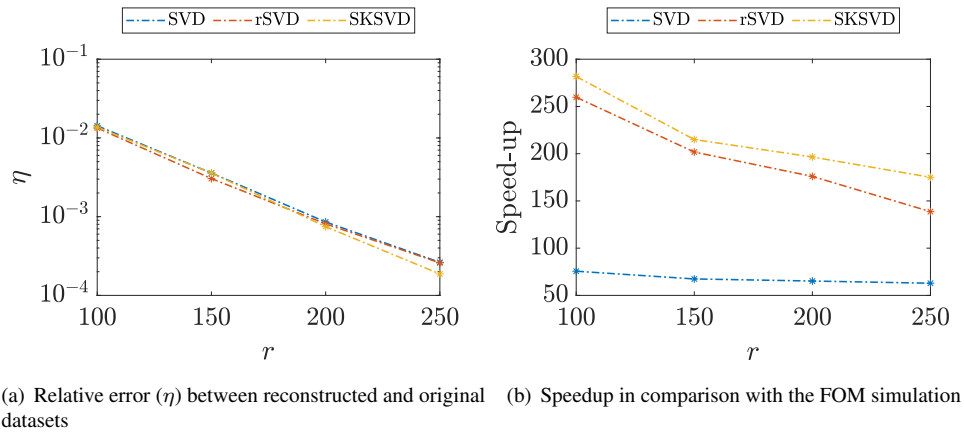


Figure 2. Accuracy and performance evaluations of the DMD code for each of the three SVD algorithms.

We also compare a snapshot from the original dataset with the DMD reconstruction. Figure 3 shows the reconstruction of the concentration field at $t = 15$ s with 100 and 250 modes, that is, our worst and best reconstructions in terms of relative error according to Figure 2. We observe that the reconstruction for $r = 100$ reveals some spurious oscillations while the solution with 250 modes has an excellent agreement with the FOM. However, these oscillations do not largely affect the shape, size, or position of the vortices generated by the Kelvin-Helmholtz instability and the front position. Furthermore, we postprocess the results and evaluate the ROM accuracy for two different QoIs, mass conservation, and the front position. Figure 4 and Table 1 shows the results, where the full black line represents the FOM data. We note that the ROMs inherit the FOM mass conservation property despite the reduced number of Koopman Modes used. Furthermore, the surrogate models provide an excellent representation of the front position x_f even for $r = 100$. We also observe from Table 1 that the ROMs results approach the FOM data for an increasing number of Koopman Modes.

Table 1. Relative error for the quantities of interest

Koopman Modes	Mass conservation	Front position
100	7.2435×10^{-4}	3.1757×10^{-3}
150	6.9677×10^{-4}	2.9271×10^{-3}
200	6.1054×10^{-5}	1.9765×10^{-3}
250	2.3256×10^{-5}	1.2555×10^{-3}

We extend our analysis and investigate the use of DMD to generate accurate representations for a different case. For the width of the heavy fluid column $L_x^0 = 1.2$ m, we use the $r = 100$ previously computed Koopman

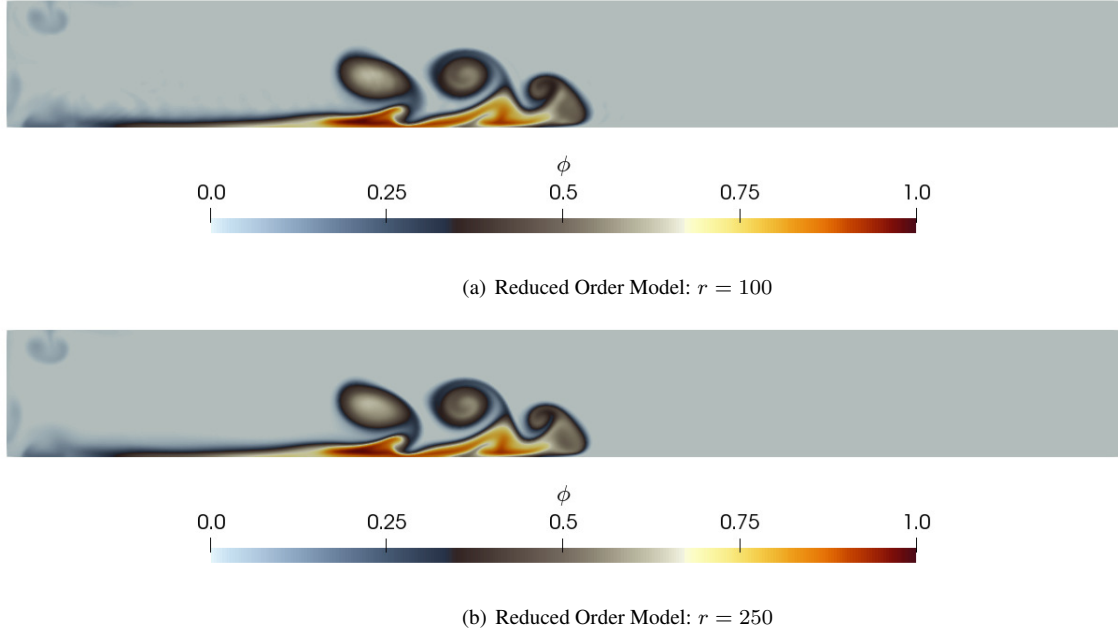


Figure 3. Concentration field at $t = 15$ s. Solution for $r = 250$ is indistinguishable from the FOM.

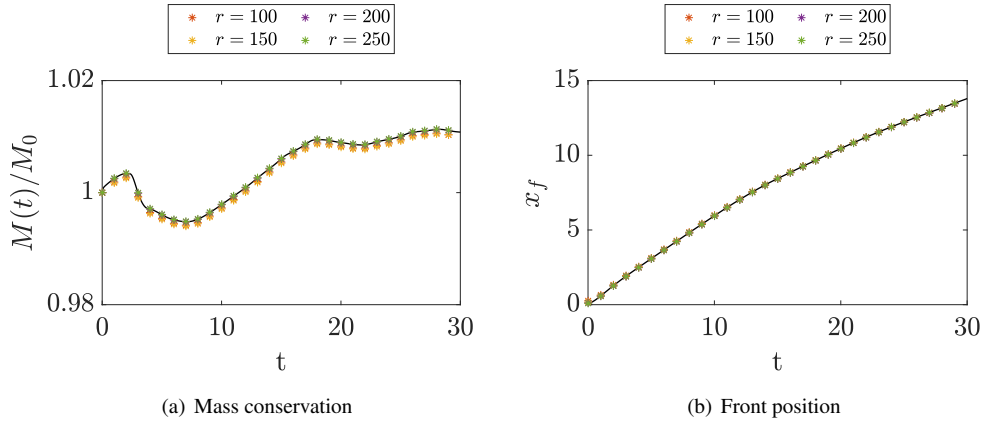


Figure 4. Comparison of quantities of interest for different values of r .

Modes to predict the new concentration field. The initial condition for this case is projected into the reduced subspace by projecting $\mathbf{b} = \Psi^\dagger \mathbf{y}_0$. Results regarding the quantities of interest are seen in Figure 5, where we compare the results predicted by the ROM with the corresponding FOM. We observe from both pictures that, even though the basis is constructed from data collected from a different simulation, the ROM can predict the FOM behavior with reasonable accuracy for both quantities of interest. The relative errors for mass conservation and front position are 2.641×10^{-3} and 2.956×10^{-2} , respectively. In terms of efficiency, we note a speedup of approximately 1273, that is, the ROM can be solved 1273 times for one FOM solving.

We also analyze the formation of the Kelvin-Helmholtz vortices in the ROM. Figure 6 shows the solution of the FOM and the ROM at $t = 30$ s and we note that the ROM is able to predict the flow physics. Table 2 compares the position of the vortices for both cases and we note that, despite of the dimensionality reduction, the formation of the vortices and their position are predicted with reasonable accuracy.

4 Conclusions

In the present work, we evaluated the effects of surrogate models created by the Dynamic Mode Decomposition method on a finite element simulation of complex density-driven flows. We consider only the concentration field

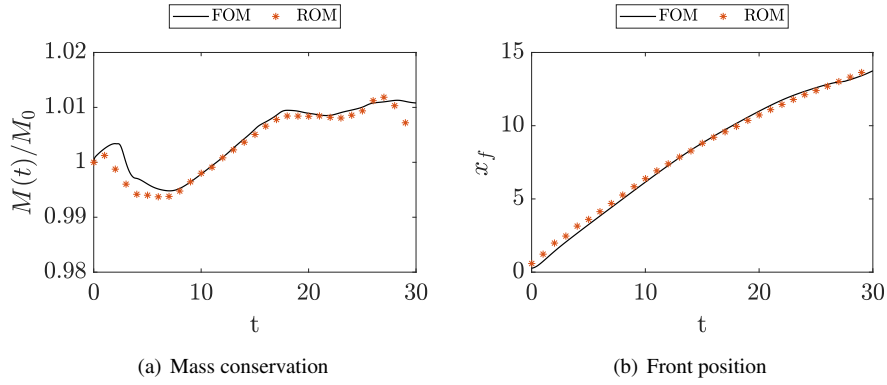
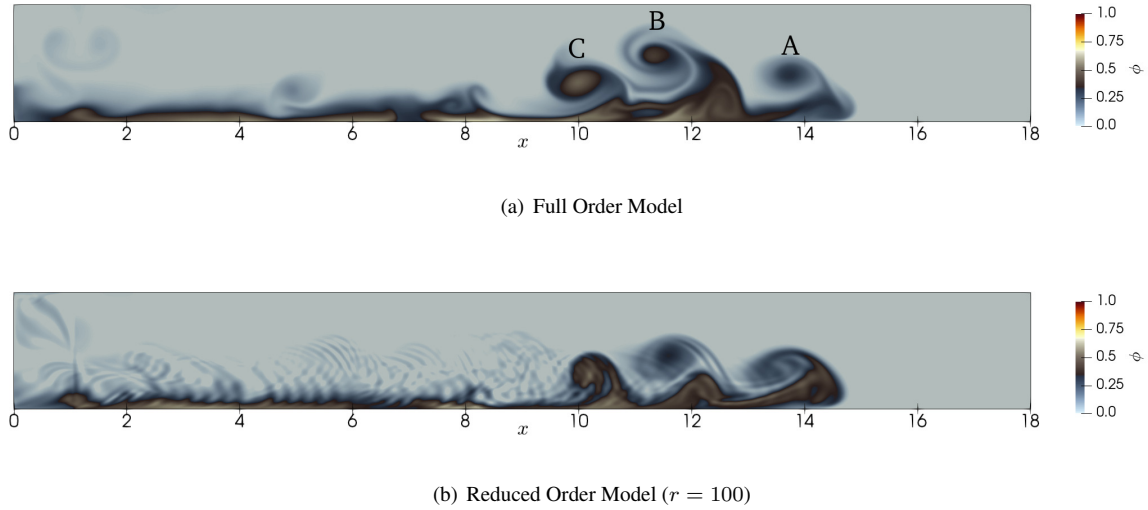

 Figure 5. ROM and FOM results for the quantities of interest ($L_x^0 = 1.2\text{m}$).

 Figure 6. FOM and ROM solutions at $t = 30\text{s}$ ($L_x^0 = 1.2\text{m}$).

 Table 2. Positions of vortices A, B and C at $t = 30\text{s}$, $L_x^0 = 1.2\text{m}$

Vortex	Full Order Model	Reduced Order Model
A	13.78	13.60
B	11.37	11.59
C	10.03	10.15

for the snapshots dataset and construct our surrogate models from 25% of the available data. We extract different numbers of Koopman Modes from the data and reconstruct the original signal with different SVD algorithms to test their performance and accuracy. We observe that the two randomized SVD algorithms present a better computational performance than the standard SVD code with negligible accuracy differences. Furthermore, we post-process our FOM and ROM data to evaluate the impact of dimensionality reduction on the usual quantities of interest in this problem. We notice that all the tested surrogate models reveal a similar behavior regarding the FOM, with better relative error compared with the prescribed error of the reprojected solutions. Even if the reconstructed solutions do not achieve a given relative error tolerance, the quantities of interest are well approximated. We extend our analysis and use the DMD to predict the evolution of the concentration field for a different simulation. We compared the obtained solutions with the full order model relative to the new configuration and noted that the

DMD could predict the flow dynamics even with a relatively small number of Koopman modes.

Acknowledgements. This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior- Brasil (CAPES) - Finance Code 001. This work is also partially supported by CNPq, FAPERJ, ANP, and Petrobras.

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] Le Clainche, S. & Vega, J. M., 2018. Analyzing Nonlinear Dynamics via Data-Driven Dynamic Mode Decomposition-Like Methods. *Complexity*, vol. Article ID 6920783, pp. 1–21.
- [2] Tu, J. H., Rowley, C. W., Luchtenburg, D. M., Brunton, S. L., & Kutz, J. N., 2014. On dynamic mode decomposition: Theory and applications. *Journal of Computational Dynamics*, vol. 1, n. 2, pp. 391–421.
- [3] Brunton, S. L., Noack, B. R., & Koumoutsakos, P., 2020. Machine Learning for Fluid Mechanics. *Annual Review of Fluid Mechanics*, vol. 52, pp. 477–508.
- [4] Willcox, K., 2007. 11. Model Reduction for Large-Scale Applications in Computational Fluid Dynamics. In *Real-Time PDE-Constrained Optimization*, pp. 217–232.
- [5] Carlberg, K., Farhat, C., Cortial, J., & Amsallem, D., 2013. The GNAT method for nonlinear model reduction: Effective implementation and application to computational fluid dynamics and turbulent flows. *Journal of Computational Physics*, vol. 242, pp. 623–647.
- [6] Willcox, K. & Peraire, J., 2002. Balanced model reduction via the proper orthogonal decomposition. *AIAA Journal*, vol. 40, n. 11, pp. 2323–2330.
- [7] Pitton, G. & Rozza, G., 2017. On the Application of Reduced Basis Methods to Bifurcation Problems in Incompressible Fluid Dynamics. *Journal of Scientific Computing*, vol. 73, pp. 157–177.
- [8] Calmet, H., Pastrana, D., Lehmkühl, O., Yamamoto, T., Kobayashi, Y., Tomoda, K., Houzeaux, G., & Vázquez, M., 2020. Dynamic Mode Decomposition Analysis of High-Fidelity CFD Simulations of the Sinus Ventilation. *Flow, Turbulence and Combustion*, vol. 105, pp. 699–713.
- [9] Rowley, C. W., Mezi, I., Bagheri, S., Schlatter, P., & Henningson, D. S., 2009. Spectral analysis of nonlinear flows. *Journal of Fluid Mechanics*, vol. 641, pp. 115–127.
- [10] Kutz, J. N., Brunton, S. L., Brunton, B. W., & Proctor, J. L., 2016. *Dynamic Mode Decomposition*. SIAM.
- [11] Schmid, P. J., 2010. Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics*, vol. 656, pp. 5–28.
- [12] Tezzele, M., Demo, N., Stabile, G., Mola, A., & Rozza, G., 2020. Enhancing CFD predictions in shape design problems by model and parameter space reduction. *arXiv 2001.05237*.
- [13] Alla, A. & Kutz, J. N., 2017. Nonlinear model order reduction via dynamic mode decomposition. *SIAM Journal on Scientific Computing*, vol. 39, n. 5, pp. B778–B796.
- [14] Halko, N., Martinsson, P. G., & Tropp, J. A., 2011. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, vol. 53, n. 2, pp. 217–288.
- [15] Brand, M., 2006. Fast low-rank modifications of the thin singular value decomposition. *Linear Algebra and Its Applications*, vol. 415, n. 1, pp. 20–30.
- [16] Van Der Walt, S., Colbert, S. C., & Varoquaux, G., 2011. The numpy array: a structure for efficient numerical computation. *Computing in Science & Engineering*, vol. 13, n. 2, pp. 22.
- [17] Necker, F., Härtel, C., Kleiser, L., & Meiburg, E., 2002. High-resolution simulations of particle-driven gravity currents. *International Journal of Multiphase Flow*, vol. 28, n. 2, pp. 279–300.
- [18] Alnæs, M. S., Blechta, J., Hake, J., Johansson, A., Kehlet, B., Logg, A., Richardson, C., Ring, J., Rognes, M. E., & Wells, G. N., 2015. The fenics project version 1.5. *Archive of Numerical Software*, vol. 3, n. 100.
- [19] Guerra, G. M., Zio, S., Camata, J. J., Rochinha, F. A., Elias, R. N., Paraizo, P. L., & Coutinho, A. L., 2013. Numerical simulation of particle-laden flows by the residual-based variational multiscale method. *International Journal for Numerical Methods in Fluids*, vol. 73, n. 8, pp. 729–749.
- [20] Peherstorfer, B. & Willcox, K., 2015. Dynamic data-driven reduced-order models. *Computer Methods in Applied Mechanics and Engineering*, vol. 291, pp. 21–41.
- [21] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E., 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830.