Finite Elements in Analysis and Design 228 (2024) 104050

Contents lists available at ScienceDirect

Finite Elements in Analysis & Design

journal homepage: www.elsevier.com/locate/finel

A comparison of data-driven reduced order models for the simulation of mesoscale atmospheric flow

Arash Hajisharifi [a], Michele Girfoglio [a], Annalisa Quaini [b,*], Gianluigi Rozza [a]

[a] mathLab, Mathematics Area, SISSA, via Bonomea 265, I-34136 Trieste, Italy
[b] Department of Mathematics, University of Houston, Houston TX 77204, USA

ARTICLE INFO

Keywords:
Proper orthogonal decomposition
Dynamic mode decomposition
Physical parametrization
Data-driven reduced order models
Atmospheric flows

ABSTRACT

The simulation of atmospheric flows by means of traditional discretization methods remains computationally intensive, hindering the achievement of high forecasting accuracy in short time frames. In this paper, we apply three reduced order models that have successfully reduced the computational time for different applications in computational fluid dynamics while preserving accuracy: Dynamic Mode Decomposition (DMD), Hankel Dynamic Mode Decomposition (HDMD), and Proper Orthogonal Decomposition with Interpolation (PODI). The three methods are compared in terms of computational time and accuracy in the simulation of two well-known 2D benchmarks for mesoscale flow. The accuracy of the DMD and HDMD solutions deteriorates rather quickly as the forecast time window expands, although these methods are designed to predict the dynamics of a system. The reason is likely the strong nonlinearity in the benchmark flows. The PODI solution is accurate for the entire duration of the time interval of interest thanks to the use of interpolation with radial basis functions. This holds true also when the model features a physical parameter expected to vary in a given range, as is typically the case in weather prediction, and for preliminary results in 3D.

1. Introduction

Despite a continuous increase in computational power, simulations of atmospheric flows using classical discretization methods (e.g., finite element methods or finite volume methods) remain computationally expensive. Given the large number of simulations required to quantify uncertainty in weather prediction, alternatives to such discretization methods, also called Full Order Models (FOMs), are needed to reduce the computational time and allow for improved prediction accuracy in short time frames.

For over a couple of decades, Reduced Order Models (ROMs) have emerged a methodology of choice to reduce the computational burden when FOM simulations have to be carried out for several (physical) parameter values, as in the case of uncertainty quantification, or for long periods of time, as in the case of forecasts. ROMs replace the FOM of choice with a lower-dimensional approximation that captures the essential behavior of the system. This is achieved through a two-step procedure. In the first step, called *offline phase*, one constructs a database of several FOM solutions associated to given times and/or physical parameter values. An example of a physical parameter for an atmospheric flow problem could be an initial temperature perturbation with magnitude expected to vary in a given range. The database of FOM solutions is used to generate a reduced basis, which is (hopefully much) smaller than the high-dimensional FOM basis but still preserves the essential features of the system. In the second step, called *online phase*, one uses this reduced basis to quickly compute the solution for newly specified times and/or parameter values. Note that,

* Corresponding author.
E-mail address: quaini@math.uh.edu (A. Quaini).

https://doi.org/10.1016/j.finel.2023.104050
Received 14 July 2023; Received in revised form 14 September 2023; Accepted 21 September 2023
Available online 5 October 2023

while the offline phase is performed once and for all, the online phase is performed as many times as needed. For a comprehensive review on ROMs, the reader is referred to, e.g., [1–6].

The ROMs that have been successfully applied to fluid dynamics problems could be divided into two major categories: projection-based vs. data-driven. In general terms, projection-based ROMs project the governing equations onto the low-dimensional subspace spanned by the basis functions of the reduced basis mentioned above. In order to implement a projection-based ROM efficiently, one needs access to the source code of the FOM solver. On the other hand, data-driven ROMs rely on available data from the high-dimensional FOM system to directly learn a reduced order model without explicitly considering the underlying equations. While projection-based methods aim to preserve the governing equations of the high-dimensional system in the reduced model, data-driven methods construct the reduced order model purely from the data, i.e., they learn the relationships and patterns observed in the data. Hence, a data-driven ROM is blind to the mathematical model and treats the FOM solver as a black box. For this reason, data-driven ROMs are also called non-intrusive.

In this paper, we focus on data-driven ROMs. The efficiency of projection-based ROMs is rather limited in the case of nonlinear problems as these problems often require hyper-reduction techniques (see, e.g., [7,8]) that are problem-dependent and computationally expensive. So, non-intrusive ROMs are to be preferred for applications where a high speed-up is required. Since atmospheric flows are highly nonlinear and high speed-up is desirable for forecasts, the aim of this paper is to compare in terms of accuracy and computational time three data driven ROMs that have been successfully applied to different fluid dynamics problems: Dynamic Mode Decomposition (DMD), Hankel Dynamic Mode Decomposition (HDMD) and Proper Orthogonal Decomposition with Interpolation (PODI). We start from DMD because it is specifically designed to predict the future behavior of a system [9–12] and it was shown to work well for problems like axisymmetric jet flow [13], annular liquid sheets [14], and turbulent cavity flow [15]. The natural next choice is HDMD because it improves the DMD algorithm with time-delay embedding [16–21]. The result is that HDMD can predict more accurately and for longer periods of time systems exhibiting strong nonlinear dynamics [20,22], as is the case of atmospheric flows. HDMD has been successfully applied to simulate, e.g., periodic cavity flow [16,23], electromechanical systems [21], and biological systems [18]. PODI differs from DMD and HDMD in that it is not designed to forecast the system evolution, but rather to interpolate solutions in a parameter space, where time is one of possibly many parameters of interest. So far, PODI has been applied to perform parametric studies for problems stemming from, e.g., thermo-mechanics [24], hemodynamics [25], chemical [26] and naval [27,28] engineering, and aeronautics [29]. This long list of successes for DMD, HDMD, and PODI, particularly in the field of computational fluid dynamics, is the reason why we have selected them for atmospheric simulations.

POD, which lies at the core of many ROMs, is often referred to as Empirical Orthogonal Function (EOF) analysis in the geophysical fluid dynamics community. It has been applied to reanalysis data to identify spatio-temporal coherent meteorological patterns and teleconnections, e.g., the Madden-Julian Oscillation, the Quasi-Biennial Oscillation, and the El Niño-Southern Oscillation. See, e.g., [30–32]. These phenomena involve large spatial and temporal scales: they result from the interaction of global circulation effects and happen over periods of time ranging from several months to many years. The EOF analysis uses data on the global scale and considers time as the only parameter. In addition, it is mostly limited to system identification. Very recently, a data-driven ROM based on EOF analysis has been used for pattern prediction, specifically to forecast the weekly average sea surface temperature [31]. Other data-driven methods borrowed from Machine Learning (ML) have been applied to global weather forecasting. See, e.g., [33–39]. However, these methods cannot be strictly categorized as reduced order modeling since no reduced basis is generated.

The work presented in this paper is the first attempt to apply ROMs that generate a reduced order basis (unlike ML methods) for both system identification and forecast of regional atmospheric flows (unlike EOF). We focus on a spatial scale of a few kilometers and a time scale of a few hours, not on the global circulation for long periods of time, with an obvious difference in resolution. Finally, with PODI we perform a parametric study that includes a physical parameter as well, i.e., our analysis is not limited to time as the only parameter.

This paper is organized as follows. Section 2 describes the compressible Euler equations for low Mach stratified flows and gives details about the full order model. Section 3 presents the main ingredients of the three ROMs under consideration. Section 4 reports the comparison of the three ROMs using three well-known benchmark problems involving stratified and gravity driven atmospheres: the rising thermal bubble in 2D and 3D and the density current. Finally, Section 5 provides conclusions and perspectives.

2. The full order model

We consider the dynamics of dry atmosphere, i.e., we neglect the effects of moisture. In addition, we neglect solar radiation and heat flux from the ground. We assume that dry air behaves like an ideal gas. Then, the equations describing the mass, momentum, and energy conservation in a spatial domain of interest Ω over a certain time interval $(0, t_f]$ are given by

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \boldsymbol{u}) = 0 \qquad \text{in } \Omega \times (0, t_f], \tag{1}$$

$$\frac{\partial (\rho \boldsymbol{u})}{\partial t} + \nabla \cdot (\rho \boldsymbol{u} \otimes \boldsymbol{u}) + \nabla p + \rho g \hat{\mathbf{k}} = \mathbf{0} \qquad \text{in } \Omega \times (0, t_f], \tag{2}$$

$$\frac{\partial (\rho e)}{\partial t} + \nabla \cdot (\rho \boldsymbol{u} e) + \nabla \cdot (p \boldsymbol{u}) = 0 \qquad \text{in } \Omega \times (0, t_f], \tag{3}$$

where ρ is the air density, $\boldsymbol{u}$ is the wind velocity, p is the pressure, $\hat{\mathbf{k}}$ is the unit vector aligned with the vertical axis z, g is the gravitational constant, and e is the total energy density. Note that $e = c_v T + |\boldsymbol{u}|^2/2 + gz$ where c_v is the specific heat capacity at

constant volume and T is the absolute temperature. To close system (1)–(3), we have the thermodynamics equation of state for ideal gases:

$$p = \rho R T, \tag{4}$$

where R is the specific gas constant of dry air.

Introducing the following splitting of the pressure

$$p = p' + \rho g z, \tag{5}$$

where $\rho g z$ is a background state and p' is a fluctuation with respect to it, Eq. (2) can be recast as:

$$\frac{\partial(\rho \boldsymbol{u})}{\partial t} + \nabla \cdot (\rho \boldsymbol{u} \otimes \boldsymbol{u}) + \nabla p' + g z \nabla \rho = 0 \qquad \text{in } \Omega \times (0, t_f]. \tag{6}$$

Let c_p be the specific heat capacity at constant pressure for dry air. By introducing the specific enthalpy $h = c_v T + p/\rho = c_p T$, the total energy can be written as $e = h - p/\rho + K + gz$, where $K = |\boldsymbol{u}|^2/2$ is the kinetic energy density. Then, Eq. (3) can be rewritten as:

$$\frac{\partial(\rho h)}{\partial t} + \nabla \cdot (\rho \boldsymbol{u} h) + \frac{\partial(\rho K)}{\partial t} + \nabla \cdot (\rho \boldsymbol{u} K) - \frac{\partial p}{\partial t} + \rho g \boldsymbol{u} \cdot \hat{\mathbf{k}} = 0 \qquad \text{in } \Omega \times (0, t_f]. \tag{7}$$

To obtain (7), we have also employed Eq. (1).

Since problem (1),(4)–(7) does not feature any dissipation mechanism, perturbations due to numerical error can lead to a simulation breakdown or large, unphysical oscillations in the computed solution. Hence, numerical stabilization is needed. Typically, it amounts to introducing additional (dissipative) terms in Eq. (6) and (7) as follows:

$$\frac{\partial(\rho \boldsymbol{u})}{\partial t} + \nabla \cdot (\rho \boldsymbol{u} \otimes \boldsymbol{u}) + \nabla p' + g z \nabla \rho - \nabla \cdot (2 \mu_a \boldsymbol{\epsilon}(\boldsymbol{u})) + \nabla \left(\frac{2}{3} \mu_a \nabla \cdot \boldsymbol{u} \right) = 0 \qquad \text{in } \Omega \times (0, t_f], \tag{8}$$

$$\frac{\partial(\rho h)}{\partial t} + \nabla \cdot (\rho \boldsymbol{u} h) + \frac{\partial(\rho K)}{\partial t} + \nabla \cdot (\rho \boldsymbol{u} K) - \frac{\partial p}{\partial t} + \rho g \boldsymbol{u} \cdot \hat{\mathbf{k}} - \nabla \cdot \left(\frac{\mu_a}{Pr} \nabla h \right) = 0 \qquad \text{in } \Omega \times (0, t_f], \tag{9}$$

where μ_a is an artificial viscosity, $\boldsymbol{\epsilon}(\boldsymbol{u}) = (\nabla \boldsymbol{u} + (\nabla \boldsymbol{u})^T)/2$ is the strain-rate tensor, and Pr is the Prandtl number, i.e., the dimensionless number defined as the ratio of momentum diffusivity to thermal diffusivity. Typically, the introduction of artificial viscosity μ_a serves the dual purpose of achieving stabilization and Large Eddy Simulation (LES). See, e.g., [40–42].

The full order model in this paper is given by the stabilized Euler equations in the formulation (1), (4), (5), (8), (9).

A quantity of interest for atmospheric problems is the potential temperature θ. Many authors choose to formulate the Euler equations with θ as a variable. Instead, we compute it from T and p using the following definition:

$$\theta = \frac{T}{\pi}, \quad \pi = \left(\frac{p}{p_0} \right)^{\frac{R}{c_p}}, \tag{10}$$

where $p_0 = 10^5$ Pa, which is the atmospheric pressure at the ground. This choice is due to implementation reasons explained in [41]. Let us also define the potential temperature fluctuation θ', which is the difference between θ and its typical hydrostatic value θ_0:

$$\theta'(x, y, z, t) = \theta(x, y, z, t) - \theta_0(z). \tag{11}$$

See, e.g., [43] for more details.

2.1. Some details about the full order method (FOM)

If one is not careful in designing an efficient numerical scheme, a solver for the full order model (1), (4), (5), (8), (9) could be computationally intensive. In order to contain the computational cost, we use a splitting scheme thoroughly described in [41]. Here, we report only some details.

To discretize in time, we introduce a time step $\Delta t \in \mathbb{R}$ to partition time interval $(0, t_f]$ and obtain time levels $t^n = t_0 + n\Delta t$, with $n = 0, \dots, N_{tf}$ and $t_f = 0 + N_{tf} \Delta t$. For the discretization of the Eulerian time derivatives in (1), (8), and (9), we adopt the Backward Euler scheme. In Eq. (8) and (9), the treatment of the convective terms is semi-implicit while the treatment of the diffusive terms is implicit. On the other hand, Eq. (1) is treated explicitly. For the space discretization, the computational domain Ω is partitioned into cells or control volumes Ω_i, with $i = 1, \dots, N_c$, where N_c is the total number of cells in the mesh. We adopt second-order finite volume schemes. Finally, in order to decouple the computation of the pressure from the computation of the velocity we use the PISO algorithm [44–46].

This FOM is implemented within GEA (Geophysical and Environmental Applications) [40,41,47,48], an open-source package for atmosphere and ocean modeling based on the finite volume C++ library OpenFOAM®.

We would like to point out that, although we have made specific choices for the full order method (e.g., spatial discretization via a finite volume method), the conclusions that we will draw about the reduced order methods presented in the next section are expected to hold also for different full order methods (e.g., finite element methods).

3. The reduced order model

Let us assume that the PDE problem described in Section 2 depends on some physical parameters that vary over a certain interval. Let d be the number of parameters of interest and $\boldsymbol{\pi}$ the vector that stores them. In addition, let $\mathcal{P} \subset \mathbb{R}^d$ be parameter space with $\boldsymbol{\pi} \in \mathcal{P}$. Although the time t could be treated as a parameter, we do not store it in $\boldsymbol{\pi}$ and deal with it separately.

The basic assumption of ROM for a PDE problem depending on time t and parameter vector $\boldsymbol{\pi}$ is that any solution can be represented in terms of a linear combination of a reduced number of global basis functions, that depend exclusively on space $\boldsymbol{x}$, with the weights of the linear combination depending only on t and $\boldsymbol{\pi}$. In the case of the potential temperature perturbation θ', which is our variable of interest, this is written as:

$$\theta'(\boldsymbol{x}; t, \boldsymbol{\pi}) \approx \theta'_r(\boldsymbol{x}; t, \boldsymbol{\pi}) = \sum_{i=1}^{N_{\theta'}} \alpha_i(t, \boldsymbol{\pi}) \varphi_i(\boldsymbol{x}), \tag{12}$$

where θ'_r is the reduced order approximation of θ', $N_{\theta'}$ is the number of basis functions, the φ_i are the basis functions and the α_i are the weights of the linear combination. If the time t is the only parameter of interest, then Eq. (12) becomes

$$\theta'(\boldsymbol{x}; t) \approx \theta'_r(\boldsymbol{x}; t) = \sum_{i=1}^{N_{\theta'}} \alpha_i(t) \varphi_i(\boldsymbol{x}). \tag{13}$$

While we focus on approximating θ', approximations similar to (12) and (13) can be applied to any other variable of interest, either primal (i.e., unknown of the original problem, density, velocity, pressure and the specific enthalpy) or derived (like θ' itself).

In this paper, we focus on three data-driven ROMs that have not been applied to mesoscale atmospheric flows yet: Dynamic Mode Decomposition (DMD), Hankel DMD (HDMD), and Proper Orthogonal Decomposition with Interpolation (PODI). All these methods rely on the Singular Value Decomposition (SVD) algorithm as a main tool to compute the basis functions φ_i. DMD and Hankel DMD are mainly designed to predict the future behavior of a system, so they are intrinsically suited for problems where the time is the parameter of interest. On the other hand, PODI relies primarily on interpolation procedures in the parameter space, so it is not designed to forecast the time evolution of the system but it interpolates solutions dependent on time and parameter $\boldsymbol{\pi}$ alike.

In the three subsections below, we report the main ingredients of the DMD and HDMD algorithms, and the PODI approach.

3.1. Dynamic mode decomposition

Introduced in [10], DMD is a useful tool to effectively extract the dominant dynamic flow structure from a unsteady flow field [11,13,49,50]. In particular, DMD forecasts future states of a non-linear time-dependent system through a linear combination of few main structures evolving linearly. Such a feature makes the DMD appealing for atmospheric problems and weather forecasts. In this work, we will adopt DMD to reconstruct, and more importantly, forecast the time evolution of θ' through (13).

In this section, we briefly present the main ingredients of the DMD algorithm concerning the computation of the basis functions φ_i and of the weights α_i in (13). For more details, we refer the reader to, e.g., [9,12,51].

Recall that N_c denotes the number of degrees of freedom of the potential temperature perturbation in the full order method. Let $\theta'_h(\boldsymbol{x}; t^i) \in \mathbb{R}^{N_c}$, with $i = 1, \dots, N_t$, be the full order solution (also called snapshot) computed at time instant t^i. As mentioned above, DMD is designed to predict the future of a system, thus the collection of the snapshots stops before the end of the time interval of interest and DMD will complete the simulation. This means that $N_t < N_{tf}$. The key idea of DMD is that there exists a governing operator matrix, denoted as $\boldsymbol{A}$, that maps $\theta'_h(\boldsymbol{x}, t^i)$ to $\theta'_h(\boldsymbol{x}, t^{i+1})$ and approximates the nonlinear dynamics of the dynamical system of interest, i.e.:

$$\theta'_h(\boldsymbol{x}, t^{i+1}) = \boldsymbol{A}\theta'_h(\boldsymbol{x}, t^i). \tag{14}$$

To find $\boldsymbol{A} \in \mathbb{R}^{N_c \times N_c}$, we build two snapshot matrices in $\mathbb{R}^{N_c \times (N_t - 1)}$:

$$\boldsymbol{S}_1 = [\theta'_h(\boldsymbol{x}, t^1), \dots, \theta'_h(\boldsymbol{x}, t^{N_t - 1})] \quad \text{and} \quad \boldsymbol{S}_2 = [\theta'_h(\boldsymbol{x}, t^2), \dots, \theta'_h(\boldsymbol{x}, t^{N_t})]. \tag{15}$$

Notice that $\boldsymbol{S}_1$ contains the first $N_t - 1$ snapshots, while $\boldsymbol{S}_2$ contains the last $N_t - 1$ snapshots. By rewriting (14) in matrix form, we obtain:

$$\boldsymbol{S}_2 = \boldsymbol{A}\boldsymbol{S}_1. \tag{16}$$

Then, we perform the SVD of the matrix $\boldsymbol{S}_1$:

$$\boldsymbol{S}_1 = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T, \tag{17}$$

where $\boldsymbol{U} \in \mathbb{R}^{N_c \times N_c}$ is the orthogonal matrix whose columns are the left singular vectors, $\boldsymbol{\Sigma} \in \mathbb{R}^{N_c \times (N_t - 1)}$ is the rectangular diagonal matrix containing the singular values $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{\min\{N_c, N_t - 1\}} \geq 0$, and $\boldsymbol{V} \in \mathbb{R}^{(N_t - 1) \times (N_t - 1)}$ is the orthogonal matrix whose columns are the right singular vectors. Symbol ^{T} denotes the transpose.

In order to form a basis for the reduced space, we adopt the POD algorithm [24,26–29]. Let $R \leq \min\{N_c, N_t - 1\}$ be the rank of the matrix $\boldsymbol{S}_1$. The POD space is spanned by the first R columns of the matrix $\boldsymbol{U}$. Then, the reduced space is constructed by

retaining the first $N_{\theta'} < R$ columns, called POD modes. The value of $N_{\theta'}$ is commonly chosen to reach a user-provided threshold δ for the cumulative energy of the singular values:

$$E = \frac{\sum_{i=1}^{N_{\theta'}} \sigma_i}{\sum_{i=1}^{R} \sigma_i} \geq \delta. \tag{18}$$

Once we have $N_{\theta'}$, we can introduce $\boldsymbol{U}_{N_{\theta'}} \in \mathbb{R}^{N_c \times N_{\theta'}}$, which comes from retaining the first $N_{\theta'}$ columns of $\boldsymbol{U}$, $\boldsymbol{\Sigma}_{N_{\theta'}} \in \mathbb{R}^{N_{\theta'} \times N_{\theta'}}$, which comes from keeping the first $N_{\theta'}$ columns and rows of $\boldsymbol{\Sigma}$, and $\boldsymbol{V}_{N_{\theta'}} \in \mathbb{R}^{(N_t-1) \times N_{\theta'}}$, which is obtained from the first $N_{\theta'}$ columns of $\boldsymbol{V}$. Then, matrix $\boldsymbol{S}_1$ is approximated as follows:

$$\boldsymbol{S}_1 \approx \boldsymbol{U}_{N_{\theta'}} \boldsymbol{\Sigma}_{N_{\theta'}} \boldsymbol{V}_{N_{\theta'}}^T. \tag{19}$$

By plugging (19) into (16), $\boldsymbol{A}$ can be approximated as:

$$\boldsymbol{A} \approx \boldsymbol{S}_2 \boldsymbol{V}_{N_{\theta'}} \boldsymbol{\Sigma}_{N_{\theta'}}^{-1} \boldsymbol{U}_{N_{\theta'}}^T. \tag{20}$$

This approximation is projected onto the POD modes to get the reduced operator matrix $\boldsymbol{A}_{N_{\theta'}} \in \mathbb{R}^{N_{\theta'} \times N_{\theta'}}$:

$$\boldsymbol{A}_{N_{\theta'}} = \boldsymbol{U}_{N_{\theta'}}^T \boldsymbol{A} \boldsymbol{U}_{N_{\theta'}} = \boldsymbol{U}_{N_{\theta'}}^T \boldsymbol{S}_2 \boldsymbol{V}_{N_{\theta'}} \boldsymbol{\Sigma}_{N_{\theta'}}^{-1} \boldsymbol{U}_{N_{\theta'}}^T \boldsymbol{U}_{N_{\theta'}} = \boldsymbol{U}_{N_{\theta'}}^T \boldsymbol{S}_2 \boldsymbol{V}_{N_{\theta'}} \boldsymbol{\Sigma}_{N_{\theta'}}^{-1}. \tag{21}$$

Next, we perform the eigendecomposition of $\boldsymbol{A}_{N_{\theta'}}$, i.e. we solve the following eigenvalue problem:

$$\boldsymbol{A}_{N_{\theta'}} \boldsymbol{W} = \boldsymbol{W} \boldsymbol{\Lambda}, \tag{22}$$

where $\boldsymbol{\Lambda} \in \mathbb{R}^{N_{\theta'} \times N_{\theta'}}$ is the diagonal matrix containing the eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_{N_{\theta'}} \geq 0$ of $\boldsymbol{A}_{N_{\theta'}}$ and $\boldsymbol{W} \in \mathbb{R}^{N_{\theta'} \times N_{\theta'}}$ is the matrix whose columns are the eigenvectors of $\boldsymbol{A}_{N_{\theta'}}$. The eigenvalues of $\boldsymbol{A}_{N_{\theta'}}$ are an approximation of the first (when arranged by magnitude) $N_{\theta'}$ eigenvalues of $\boldsymbol{A}$ and are known as DMD eigenvalues [10,52].

As basis functions φ_i in (13), one takes the eigenvectors of $\boldsymbol{A}$ associated to the DMD eigenvalues. It can be shown [12] that these eigenvectors, called exact DMD modes, are the columns of the matrix $\boldsymbol{\varphi} \in \mathbb{R}^{N_c \times N_{\theta'}}$:

$$\boldsymbol{\varphi} = \boldsymbol{S}_2 \boldsymbol{V}_{N_{\theta'}} \boldsymbol{\Sigma}_{N_{\theta'}}^{-1} \boldsymbol{W}. \tag{23}$$

Another possibility (not explored in this paper) is to take the columns of $\boldsymbol{U}_{N_{\theta'}} \boldsymbol{W}$, which are known as projected DMD modes [10].

Finally, the temporal coefficients α_i in Eq. (13) are given by

$$\alpha_i(t) = e^{\lambda_i t}, \quad i = 1, \ldots, N_{\theta'}. \tag{24}$$

3.2. Hankel dynamic mode decomposition

The Hankel DMD [16,17,20,21] is a variation of the standard DMD algorithm based on the idea to combine the DMD algorithm with time delay-embedding [16,18,19]. In delay embedding, a given time-dependent datum, which in our case is the snapshot $\theta'_h(\boldsymbol{x}, t)$, is augmented by a time history of previous data.

Let M be the embedding dimension and let $\boldsymbol{H}(\theta'_h(\boldsymbol{x}, t^i)) \in \mathbb{R}^{N_c \times M}$, with $i = 1, \ldots, N_t$ and $N_t < N_{tf}$, be the so-called Hankel matrix associated to the snapshot $\theta'_h(\boldsymbol{x}, t^i)$:

$$\boldsymbol{H}(\theta'_h(\boldsymbol{x}, t^i)) = [\theta'_h(\boldsymbol{x}, t^i), \theta'_h(\boldsymbol{x}, t^{i-1}), \ldots \theta'_h(\boldsymbol{x}, t^{i-(M-1)})]. \tag{25}$$

We build two matrices in $\mathbb{R}^{N_c \times ((N_t - M)M)}$:

$$\begin{aligned} \boldsymbol{S}_1^H &= \left[\boldsymbol{H}(\theta'_h(\boldsymbol{x}, t^M)) \quad \boldsymbol{H}(\theta'_h(\boldsymbol{x}, t^{M+1})) \quad \ldots \quad \boldsymbol{H}(\theta'_h(\boldsymbol{x}, t^{N_t-1}))\right], \\ \boldsymbol{S}_2^H &= \left[\boldsymbol{H}(\theta'_h(\boldsymbol{x}, t^{M+1})) \quad \boldsymbol{H}(\theta'_h(\boldsymbol{x}, t^{M+2})) \quad \ldots \quad \boldsymbol{H}(\theta'_h(\boldsymbol{x}, t^{N_t}))\right]. \end{aligned} \tag{26}$$

Then, we follow the DMD algorithm described Section 3.1 by simply replacing $\boldsymbol{S}_1$ with $\boldsymbol{S}_1^H$ and $\boldsymbol{S}_2$ with $\boldsymbol{S}_2^H$.

It should be noted that the embedding dimension M is a crucial parameter affecting the accuracy of the HDMD algorithm. The optimal value of M could depend on the problem at hand and, to the best of our knowledge, there is no general rule to choose it [19,53,54]. In this work, its value is determined through a trial and error procedure, i.e., we try several values of M and choose the one that maximizes the accuracy of the associated HDMD-based ROM in the L^2 norm. See Section 4.1 for more details. Of course, such a procedure is time-consuming and leaves room to further research for improvement.

3.3. Proper orthogonal decomposition with interpolation

In the PODI method, which was introduced in [55], the POD algorithm is used to extract the reduced basis functions from the set of full order solutions associated with given values of time and $\boldsymbol{\pi}$, whilst the coefficients α_i are approximated by an interpolation technique. In the following, we briefly recall the main steps required by PODI method.

Following the notation introduced in Sections 2 and 3.1, let $\theta'_h(\boldsymbol{x}; t^i, \boldsymbol{\pi}^j) \in \mathbb{R}^{N_c}$, with $i = 1, \dots, N_{tf}$ and $j = 1, \dots, N_\pi$, be the full order solution computed at time instant t^i and for parameter value $\boldsymbol{\pi}^j$. We set $N_s = N_{tf} \cdot N_\pi$ and arrange these full order solutions as the columns of the snapshot matrix

$$\boldsymbol{S} = [\theta'_h(\boldsymbol{x}; t^1, \boldsymbol{\pi}^1), \theta'_h(\boldsymbol{x}; t^2, \boldsymbol{\pi}^1), \dots, \theta'_h(\boldsymbol{x}; t^{N_{tf}}, \boldsymbol{\pi}^{N_k})] \in \mathbb{R}^{N_c \times N_s}. \tag{27}$$

The SVD of matrix $\boldsymbol{S}$ gives us:

$$\boldsymbol{S} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T, \tag{28}$$

where the explanation for $\boldsymbol{U} \in \mathbb{R}^{N_c \times N_c}$, $\boldsymbol{\Sigma} \in \mathbb{R}^{N_c \times N_s}$, and $\boldsymbol{V} \in R^{N_s \times N_s}$ is provided in Section 3.1. The POD space is constructed by retaining the first $N_{\theta'} \leq \min\{N_c, N_s\}$ columns of matrix $\boldsymbol{U}$. The value of $N_{\theta'}$ is set as described in Section 3.1 and, like in Section 3.1, we will denote with $\boldsymbol{U}_{N_{\theta'}}$ the matrix that contains the first $N_{\theta'}$ columns of $\boldsymbol{U}$.

Now that we have the basis functions, we can use Eq. (12) to approximate the snapshots:

$$\theta'_h(\boldsymbol{x}; t^i, \boldsymbol{\pi}^j) \approx \theta'_r(\boldsymbol{x}; t^i, \boldsymbol{\pi}^j) = \sum_{L=1}^{N_{\theta'}} \alpha_L(t^i, \boldsymbol{\pi}^j)\varphi_L(\boldsymbol{x}), \tag{29}$$

for $i = 1, \dots, N_{tf}$ and $j = 1, \dots, N_\pi$, where the coefficients $\alpha_L(t^i, \boldsymbol{\pi}^j)$ are the entries of the matrix $\boldsymbol{C} = \boldsymbol{U}_{N_{\theta'}}^T \boldsymbol{S} \in \mathbb{R}^{N_{\theta'} \times N_s}$, i.e., $\alpha_L(t^i, \boldsymbol{\pi}^j) = \boldsymbol{C}_{ij}$. Given these coefficients at t^i and $\boldsymbol{\pi}^j$, we construct an interpolant with Radial Basis Functions [56] as follows:

$$\alpha_L(t^i, \boldsymbol{\pi}^j) = \sum_{n=1}^{N_\pi} \sum_{m=1}^{N_{tf}} w_{L_{m,n}} \zeta_{L_{m,n}}(\|(t^i, \boldsymbol{\pi}^j) - (t^m, \boldsymbol{\pi}^n)\|). \tag{30}$$

Here, $\zeta_{L_{m,n}}$ are the Radial Basis Functions (chosen to be Gaussian functions) centered in $(t^m, \boldsymbol{\pi}^n)$ and $w_{L_{m,n}}$ are unknown weights. We find the weights by solving the linear system associated to (30):

$$\boldsymbol{Z}_L \boldsymbol{w}_L = \boldsymbol{\alpha}_L. \tag{31}$$

In (31), $\boldsymbol{w}_L$ is the vector obtained as follows:

$$\boldsymbol{w}_{L_k} = w_{L_{m,n}}, \quad k = (m) + (n-1) \cdot N_{tf}, \quad 1 \leq m \leq N_{tf}, \quad 1 \leq n \leq N_\pi,$$

i.e., the first N_{tf} entries of $\boldsymbol{w}_L$ are $w_{L_{1:N_{tf},1}}$, which are related to $\boldsymbol{\pi}^1$, the following N_{tf} entries are $w_{L_{1:N_{tf},2}}$, which are related to $\boldsymbol{\pi}^1$, and so on. Matrix $\boldsymbol{Z}_L$ is constructed from the $\zeta_{L_{m,n}}$ using a similar strategy.

We remark that it is done only once during the offline phase.

Let us now consider a solution that does not belong to the snapshot matrix, i.e., we want to compute $\theta'(\boldsymbol{x}; t^{new}, \boldsymbol{\pi}^{new})$ for new time t^{new} and new parameter value $\boldsymbol{\pi}^{new}$. We obtain

$$\theta'_r(\boldsymbol{x}; t^{new}, \boldsymbol{\pi}^{new}) = \sum_{L=1}^{N_{\theta'}} \alpha_L(t^{new}, \boldsymbol{\pi}^{new})\varphi_L(\boldsymbol{x}), \tag{32}$$

where coefficients $\alpha_L(t^{new}, \boldsymbol{\pi}^{new})$ are computed by:

$$\alpha_L(t^{new}, \boldsymbol{\pi}^{new}) = \sum_{i=1}^{N_{tf}} \sum_{j=1}^{N_\pi} w_{L_{m,n}} \zeta_{L_{m,n}}(\|(t^{new}, \boldsymbol{\pi}^{new}) - (t^i, \boldsymbol{\pi}^j)\|). \tag{33}$$

Notice that while (30) is used to find the weights $w_{L_{m,n}}$, (33) is used to find α_L.

4. Numerical results

To validate our ROM approaches, we consider two standard 2D benchmarks for atmospheric flows: the rising thermal bubble [42,57–59] and the density current [42,57,60–63]. Both test cases involve a perturbation of a neutrally stratified atmosphere with uniform background potential temperature. It is worth to note that there exist several variations of these benchmarks, featuring different geometries and/or initial conditions. We use the setting from [57] for the rising thermal bubble and the setting from [60,63] for the density current.

We compare the ROM techniques presented in Section 3 in terms of the reconstruction of the time evolution of the potential temperature perturbation for the rising thermal bubble in Section 4.1 and for the density current in Section 4.2. Moreover, in Section 4.2 we present a parametric study with PODI, where the varying parameter is the amplitude of the initial temperature perturbation in the density current benchmark. Finally, Section 4.3 presents preliminary results in 3D using a tree-dimensional variant of the rising thermal bubble benchmark taken from [62].

All the ROM techniques utilized in this study are implemented in open-source Python package EZyRB [64]. The reader interested in the validation of the FOM model is referred to [41].

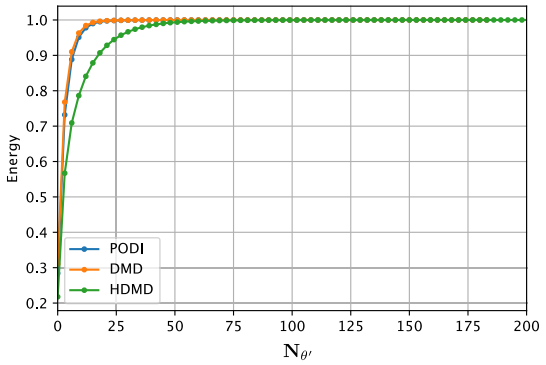

Fig. 1. Rising thermal bubble: cumulative energy E in (18) for the three ROM approaches under consideration.

4.1. Rising thermal bubble

The computational domain for this benchmark is $\Omega = [0, 5000] \times [0, 10000]$ m^2 in the xz-plane. In this domain, a neutrally stratified atmosphere with uniform background potential temperature $\theta_0 = 300$ K is perturbed by a circular bubble of warmer air. The initial temperature field is

$$\theta^0 = 300 + 2\left[1 - \frac{r}{r_0}\right] \text{ if } r \le r_0 = 2000 \text{ m}, \quad \theta^0 = 300 \text{ otherwise}, \tag{34}$$

where $(x_c, z_c) = (5000, 2000)$ m and $r = \sqrt{(x - x_c)^2 + (z - z_c)^2}$ are the center and the radius of the circular perturbation, respectively [57,58]. The initial density is given by

$$\rho^0 = \frac{p_g}{R\theta_0}\left(\frac{p}{p_g}\right)^{c_v/c_p} \quad \text{with} \quad p = p_g\left(1 - \frac{gz}{c_p\theta^0}\right)^{c_p/R}, \tag{35}$$

where $c_p = R + c_v$, with $c_v = 715.5$ J/(Kg K) and $R = 287$ J/(Kg K). The initial velocity field is zero everywhere. The initial specific enthalpy is defined as:

$$h^0 = c_p\theta^0\left(\frac{p}{p_g}\right)^{\frac{R}{c_p}}. \tag{36}$$

We impose impenetrable, free-slip boundary conditions on all the boundaries and we set $t_f = 1020$ s. In time interval $(0, 1020]$ s, the warm bubble rises due to buoyancy and evolves into a mushroom-like shape as a result of shear stress.

We generate a uniform structured mesh with mesh size $h = \Delta x = \Delta z = 62.5$ m and set the time step set to $\Delta t = 0.1$ s. Following [41,57], we set $\mu_a = 15$ and $Pr = 1$ in (8)–(9). Both of these are ad-hoc values provided in [57] to stabilize the numerical simulations. More sophisticated LES models can be found in, e.g., [40,42].

We collect an original database consisting of 204 snapshots, i.e., the computed θ' every 5 s. These snapshots are divided into two different sets. A first set, called training set, is used to generate the reduced basis. All the snapshots belonging to the training set are stored in the matrix $\boldsymbol{S}$ for PODI, while all but one are stored in $\boldsymbol{S}_1$ for DMD, and $\boldsymbol{S}_1^H$ for HDMD as explained in Sections 3.1 and 3.2. The second set, called validation set, is the complement of the training set in the original database and it is used to assess the accuracy of the ROM solution. The partitioning into these sets can be done in two ways: randomly or by preserving the temporal order (i.e., keeping the snapshots associated with the first N_t times).

Out of the 204 computed θ' in the original database, we take 184 (i.e., 90% of the database) to form the training set. In the case of DMD and Hankel DMD, these 184 solutions are the first 184 in the database (associated to the time interval $(0, 920]$ s). In the case of PODI instead, these 184 solutions are selected randomly over the entire time interval $[0, 1020]$ s. For all three methods, the remaining 20 solutions form the validation set. This difference in the training and validation sets reflects the different nature of PODI and DMD/HDMD algorithms: the former relies on the reconstruction of the solution, the latter predicts the future behavior of a system.

We start with the plot of the cumulative eigenvalues energy E (18) for PODI, DMD and HDMD in Fig. 1. We see that the curves for PODI and DMD are very close, while the curve for HDMD is farther apart. In particular, HDMD requires a larger number of singular values to reach a given energy level. To clarify the extent of this difference, Table 1 reports the number of modes needed to attain $\delta = 0.7, 0.9, 0.99$ in (18). We observe that DMD and PODI require the same number of modes for a given δ, while HDMD needs about or more than twice as many modes.

Next, Figs. 2 and 3 show a qualitative comparison of the ROM solutions for $\delta = 0.7, 0.9, 0.99$ with the FOM solution. Among the five time instants chosen for the visualization, two of them, namely $t = 255$ s and $t = 505$ s, correspond to solutions belonging to the training set. These two times allow us to asses the ability of each ROM technique to identify the system dynamics. The remaining

Table 1
Rising thermal bubble: number of modes required to retain different energy thresholds, $\delta = 0.7, 0.9, 0.99$, for the three ROM methods under consideration.

	DMD	HDMD	PODI
$\delta = 70\%$	4	7	4
$\delta = 90\%$	8	19	8
$\delta = 99\%$	17	46	17

Table 2
Rising thermal bubble: computational time needed to construct the reduced basis offline (Basis) and to perform a simulation online (Online) for DMD, HDMD, and PODI when δ is set to $0.7, 0.9, 0.99$.

δ	DMD		HDMD		PODI	
	Basis	Online	Basis	Online	Basis	Online
70%	0.0833 s	0.0182 s	2.487 s	2.73 s	0.074 s	0.012 s
90%	0.0835 s	0.0193 s	2.546 s	2.822 s	0.092 s	0.015 s
99%	0.085 s	0.02 s	2.865 s	2.95 s	0.1 s	0.018 s

three times, i.e., $t = 930, 980, 1020$ s, are not associated with the training set and thus are used to check the accuracy of the ROM in predicting (for DMD and HDMD) or interpolating (for PODI) the system dynamics. Let us discuss the results in Figs. 2 and 3 starting from the system identification. For $\delta = 0.7, 0.9$, HDMD provides a better reconstruction of θ' than PODI and DMD, whose solutions are spoiled by visible oscillations above the bubble. See the first two rows in Fig. 2. However, when δ is increased to 0.99 we observe that all three ROMs reconstruct θ' well. Indeed, no significant difference can be observed in the panels of row one and two of Fig. 3. Now, let us take a look at the solutions corresponding to the times not associated with the training set. From the bottom three rows in Figs. 2 and 3, we see that for none of the values of δ DMD can correctly predict the evolution of θ', while HDMD and PODI reconstruct well the solution when δ is set to 0.99. The problem with the reconstruction provided by DMD appears to be related to assigning large weights to basis functions associated to previous times. Indeed, in the DMD solution in Fig. 3 for $t = 1020$ s we can observe the time history of the system dynamics, i.e., the rising of the warm bubble. This will be even more evident in the DMD results for the density current benchmark (see Fig. 7).

To quantitate the agreement between the ROM solutions and the FOM solutions in Figs. 2 and 3, we report in Fig. 4 the L^2 error

$$E_{\theta'}(t) = 100 \cdot \frac{\|\theta'_h(t) - \theta'_r(t)\|_{L^2(\Omega)}}{\|\theta'_h(t)\|_{L^2(\Omega)}}, \tag{37}$$

for the three values of δ under consideration. We see that the L^2 errors for DMD and HDMD increase sharply around and after $t = 920$ s for all values of δ. Since the training set for DMD and HDMD includes only solutions before $t = 920$ s, an increase in the errors past that time is to be expected. However, the magnitude of the errors at the end of the time interval, which in the best case (HDMD with $\delta = 0.99$) is around 20%, indicates that both DMD and HDMD do a poor job in predicting the evolution of the warm bubble. This is somewhat surprising since visually the HDMD solution for $\delta = 0.99$ compares well with the FOM solution for all the times shown in Fig. 3, including the times corresponding to the validation set. We suspect that the larger error for HDMD is due to a difference in rising speed for the bubble, and not so much in the bubble shape. Since the snapshots in the training dataset for PODI were chosen randomly, and thus include FOM solutions past $t = 920$ s, the L^2 errors for PODI are comparable to (for $\delta = 0.7$) or smaller (for $\delta = 0.9, 0.99$) than the errors for DMD and HDMD when $t > 920$ s. In particular, we notice that then $\delta = 0.99$ the L^2 errors for PODI remain less than 1% for the entire duration of the time interval. In time interval $(0, 920]$ s, which corresponds to system identification, HDMD reconstructs the solution more accurately than DMD and PODI. This is especially true for $\delta = 0.7, 0.9$.

We point out that for HDMD we set $M = 25$. We tried $M = 5, 10, 25, 50, 75$ and found that this value provided a reasonable trade-off between accuracy measured with error (37) and computational cost, mentioned below.

We ran all the simulations on a 11th Gen Intel(*R*) Core(TM) i7-11700 @ 2.50 GHz system with 32 GB RAM. The time required by the FOM to complete the simulation is 65 s. Table 2 reports the computational time needed to construct the reduced basis offline and to perform a simulation online for each of the ROM we consider. The computational times for DMD and PODI are comparable, which for the online phase is explained by the fact that the DMD and PODI reduced basis have the same size for the three values of δ we consider (see Fig. 1). The speed up, i.e., the ratio between the time for a FOM simulation and the online time for the ROM, is of the order of 3000 for these two methods, specifically 65/0.02 = 3250 for DMD and 65/0.018 = 3611 for PODI. HDMD is computationally more expensive than DMD and PODI, which is the price one has to pay for the increased accuracy in system identification (see Fig. 4). The speed up for HDMD is only about 20.

4.2. Density current

The computational domain for this benchmark is $\Omega = 25600 \times 6400$ m^2 in xz-plane. Like for the previous benchmark, we start from a neutrally stratified atmosphere with uniform background potential temperature θ_0=300 K by introducing a perturbation.

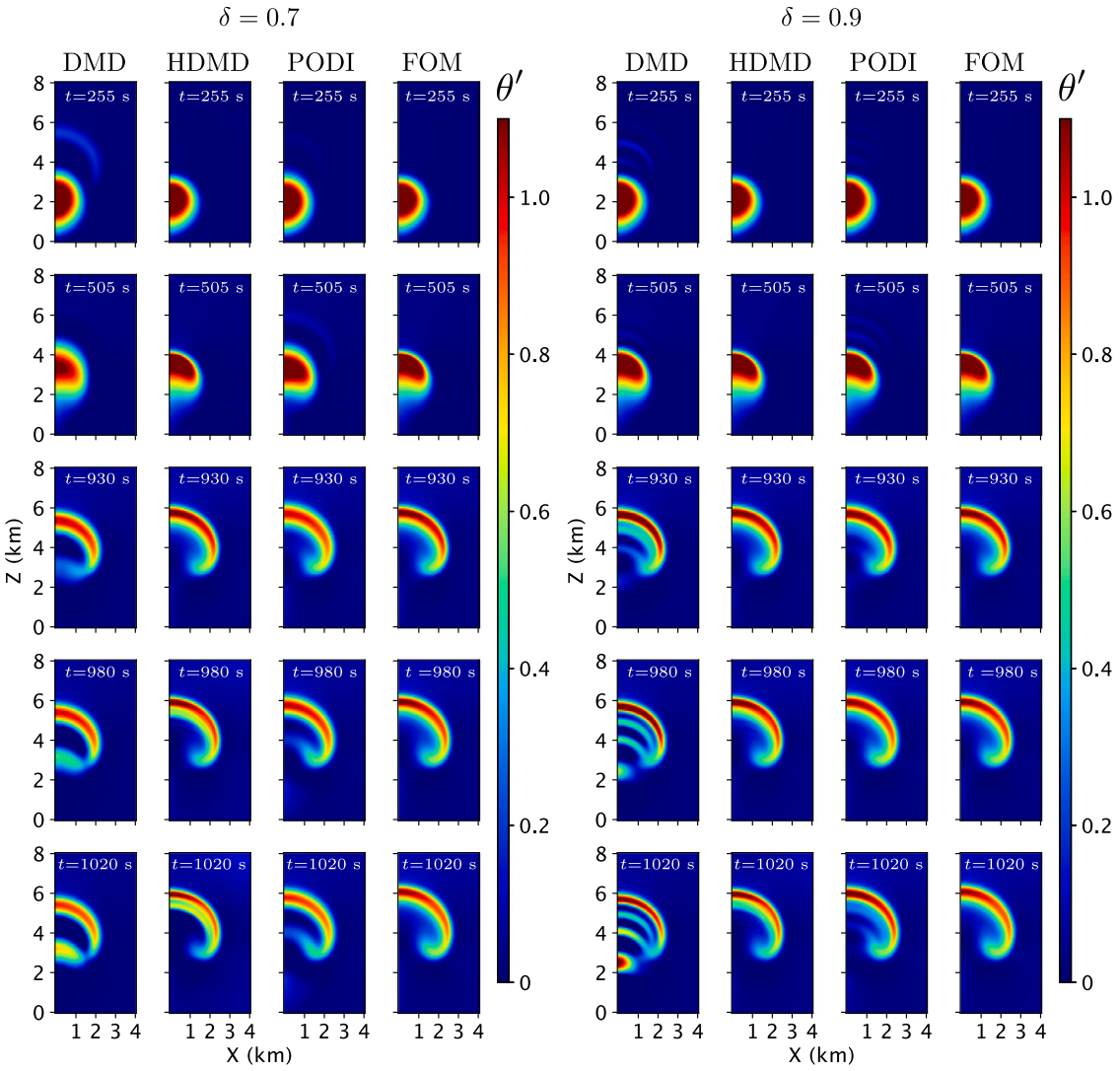

Fig. 2. Rising thermal bubble: comparison of the evolution of θ' given by the ROMs (first 3 columns in each panel) and the FOM (last column in each panel) for $\delta = 0.7$ (left panel) and $\delta = 0.9$ (right panel).

However, unlike the previous benchmark, the perturbation is represented by a circular bubble of colder air. The initial temperature field is

$$\theta_0 = 300 - \theta_s[1 + \cos(\pi r)], \quad \text{if } r \leq 1, \quad \text{otherwise } \theta_0 = 300, \tag{38}$$

where $r = \sqrt{\left(\frac{x-x_c}{x_r}\right)^2 + \left(\frac{z-z_c}{z_r}\right)^2}$, with $(x_r, z_r) = (4000, 2000)$ m and $(x_c, z_c) = (0, 3000)$ m. In (38), θ_s is the semi-amplitude of the initial temperature perturbation. In [42,57,60–63], θ_s is set to 7.5. We will start from this value as well, but later will let θ_s vary in a given interval. The initial density is given by (35), while the initial specific enthalpy is (36). The initial velocity field is zero everywhere. We impose impenetrable, free-slip boundary conditions on all the walls. In the time interval of interest, which is $(0, 900]$ s, the cold air descends due to negative buoyancy and when it reaches the ground, it rolls up and forms a front. As this front propagates, a multi-rotor structure develops. One important difference with respect to the previous benchmark is that in this test the flow structures have a predominantly vertical motion till about $t = 280$ s (fall of the cold air) and then the motion becomes predominantly horizontal (front propagation). See Fig. 5 for the initial solution and computed solution at $t = 280$ s.

We generate a uniform structured mesh with mesh size $h = \Delta x = \Delta z = 100$ m and set the time step set to $\Delta t = 0.1$ s. Following [57,63], we set $\mu_a = 75$ and $Pr = 1$ in (8)–(9). Like in the case of the previous benchmark, these are ad-hoc values used to stabilize the numerical simulations.

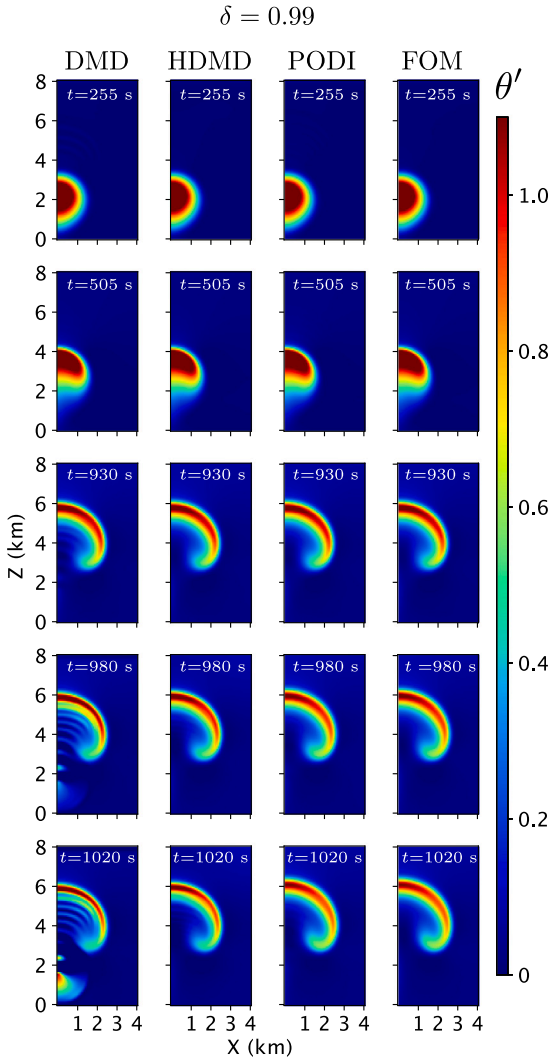

Fig. 3. Rising thermal bubble: comparison of the evolution of θ' given by the ROMs (first 3 columns) and the FOM (last column) for $\delta = 0.99$.

We will compare the ROM techniques under consideration in terms of the reconstruction of the time evolution of θ' in Section 4.2.1. In addition, in Section 4.2.2 we perform a parametric study for θ_s with PODI.

4.2.1. Time reconstruction

For the results in this section, we set $\theta_s = 7.5$. To generate the reduced basis, we collect a database consisting of 225 snapshots, i.e., the computed θ' every 4 s. Similarly to the procedure reported in Section 4.1, we divide the database into two sets: a training set containing 90% of the snapshots (i.e., 202 snapshots) and a validation set containing the remaining 10% (i.e., 23 snapshots). The partition of the database is performed sequentially for DMD and HDMD and corresponds to the first 202 snapshots. In the case of PODI, the 202 snapshots in the training set are selected randomly from the entire time interval. For HDMD, we set $M = 50$, which is larger that the value we used for the warm bubble because the density current flow is more complex.

The cumulative eigenvalue energy (18) as the number of eigenvalues is increased is shown in Fig. 6 for all the ROM approaches. We observe a less steep increase than in Fig. 1, indicating that more modes are required to capture the flow dynamics in the density current test than in the thermal rising bubble test. Table 3 displays the number of modes required to capture different energy thresholds ($\delta = 0.7, 0.9, 0.99$) for PODI, DMD, and HDMD. PODI and DMD require the same number of modes for $\delta = 0.7$ and similar numbers for $\delta = 0.9, 0.99$. The number of modes needed by HDMD is larger for every δ. While the observations for Table 3 are similar to the observation for Table 1, the numbers in Table 3 are larger. Again, this is due to the fact that the density current produces a more complex flow.

In view of the results in Section 4.1, we set the energy threshold δ to 99%. See Table 3 for the number of modes retained for the different methods. Fig. 7 compares the evolution of the potential temperature perturbation given by DMD, HDMD, and PODI with

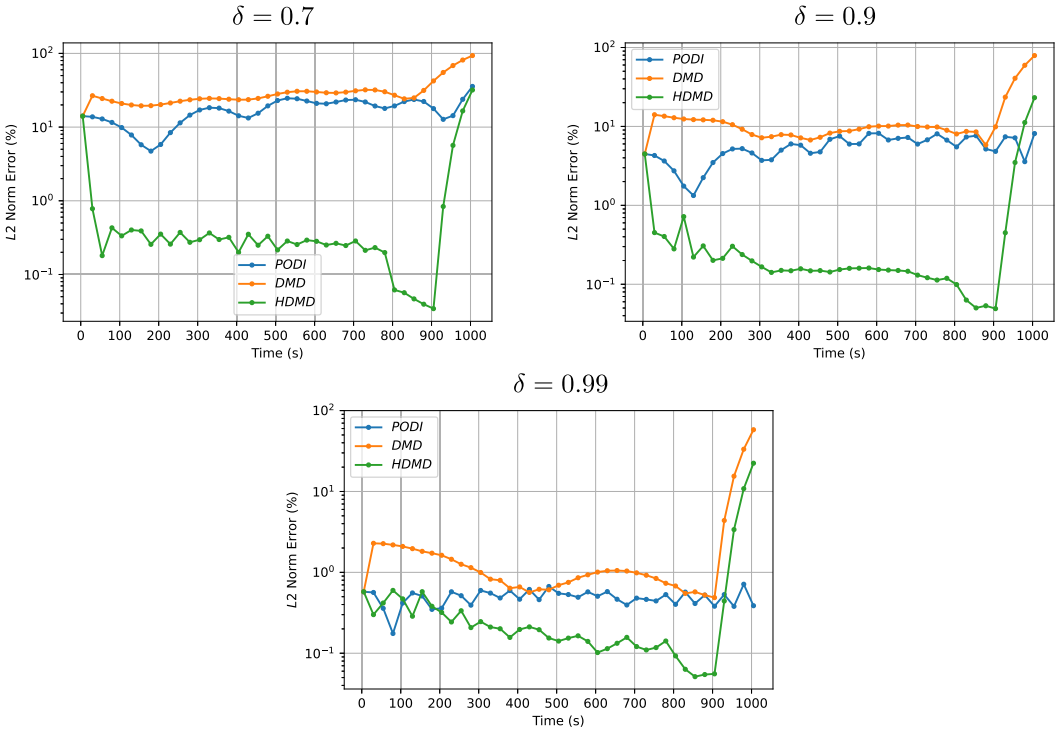

Fig. 4. Rising thermal bubble: time evolution of the L^2 error (37) for DMD, HDMD, and PODI for $\delta = 0.7$ (top left panel), $\delta = 0.9$ (top right panel) and $\delta = 0.99$ (bottom panel).

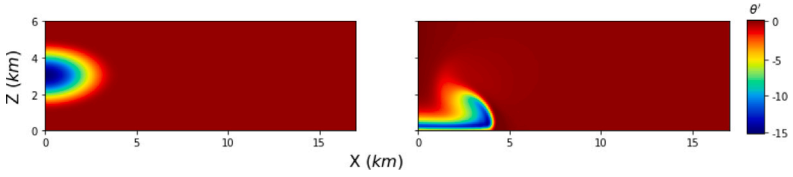

Fig. 5. Density current: initial condition (left) and computed solution at $t = 280$ s (right).

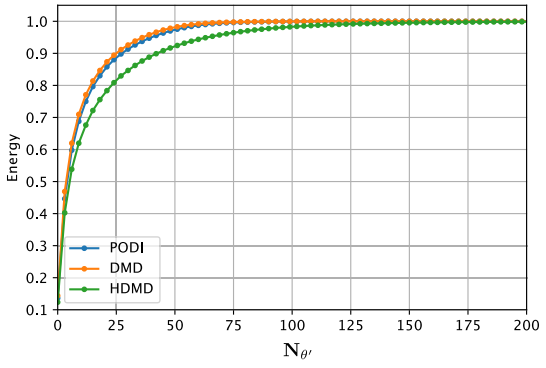

Fig. 6. Density current: cumulative energy E in (18) for the three ROM approaches under consideration.

the evolution computed by the FOM. The top two rows in Fig. 7 correspond to solutions belonging to the training set. We see that all three methods can accurately identify the main flow structure, but the DMD and PODI solutions show some instability for $x < 5$ Km and $x \in [10, 15]$ Km, respectively. These instabilities, that are especially evident at $t = 600$ s, are not present in the HDMD solutions.

Table 3
Density current: number of modes required to retain different energy thresholds, $\delta = 0.7, 0.9, 0.99$, for three ROM methods under consideration.

	DMD	HDMD	PODI
$\delta = 70\%$	11	15	11
$\delta = 90\%$	28	44	29
$\delta = 99\%$	63	120	65

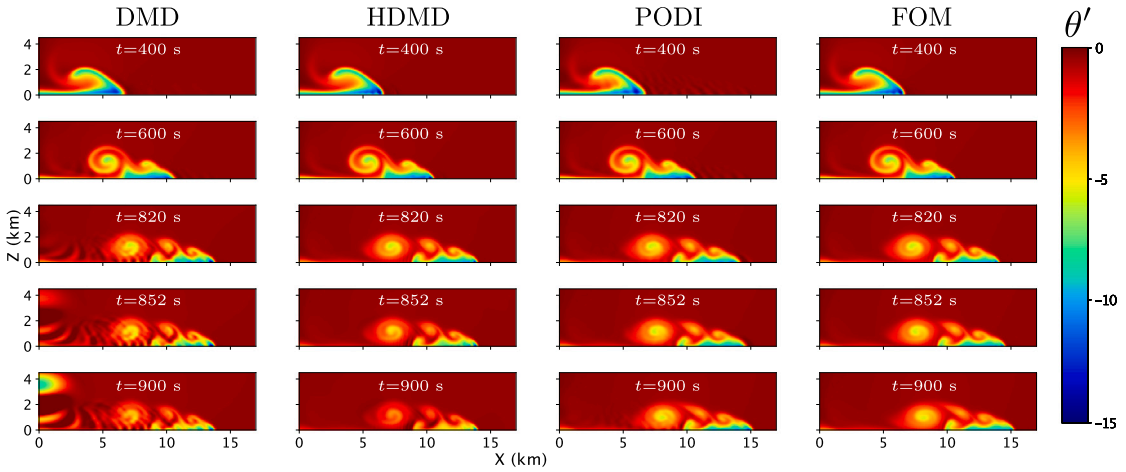

Fig. 7. Density current: comparison of evolution of θ' given by the ROMs (first 3 columns) and the FOM (last column) for $\delta = 0.99$.

The bottom three rows in Fig. 7 are not associated with the training set. We see that the instabilities in the DMD solution grow in time and expand to the majority of the domain, making the DMD prediction of the system dynamics off. As noted for the DMD results in Fig. 3, it appears that the DMD solution gives a large weight to many basis functions associated to previous times. Indeed, in the DMD solution for $t = 852$ s we can observe the time history of the system dynamics, i.e. the fall of the cold bubble and the front propagation. We note also that the more time passes, the larger the weights for the "past" snapshots become, i.e., the entire evolution of the flow structures becomes more visible. In fact, the DMD solution for $t = 900$ s clearly shows the initial cold bubble, which is not present in the DMD solution for $t = 820$ s. The HDMD algorithm fixes such issue and provides a very good prediction of θ' field for $t = 820$ s and $t = 852$ s. For $t = 900$ s, the HDMD solution compares less favorably with the FOM solution, although no instability emerges. Finally, the PODI solution suppresses the instability for $x \in [10, 15]$ Km as time passes and is accurate for $t = 820, 852$ s. For $t = 900$ s though, instabilities arise in the PODI solution for $x < 5$ Km.

For a more quantitative comparison, we show the time evolution of L^2 error (37) in Fig. 8. Although qualitatively the HDMD solution compares well with the FOM solution (See Fig. 7), we see that both DMD and HDMD do a poor job in predicting the future behavior of the system in the L^2 norm. Our suspicion that HDMD captures the shape of the flow structures well, but not their propagating speed, is confirmed with this test: the large L^2 error for HDMD is mainly due to the fact that the front propagation slows down in comparison to the FOM solution. In fact, in the FOM solution the front is located around $x = 15$ Km at $t = 900$ s, while in the HDMD solution is around $x = 14$ Km at the same time. Their accuracy of DMD and HDMD is even worse than for the previous benchmark, as it is clear from comparing Fig. 8 to the bottom panel of Fig. 4. Specifically, the L^2 error (37) for DMD increases from about 60% to around 140%, while the error for HDMD increases from 22% to about 60%. This is a further confirmation that it is more challenging to capture the flow dynamics in the density current test than the thermal rising bubble test. On the other hand, thanks to the interpolatory approach the PODI solution maintains a good accuracy throughout the entire time interval, oscillating around 1% during fall of the bubble and reaching a maximum of about 3% in the horizontal convection phase.

To acknowledge the fact that this benchmark features mainly vertical dynamics before mainly horizontal dynamics, we now restrict the database to the computed solutions for $t \geq 280$ s, i.e., we discard the snapshots associated to the mainly vertical motion. This idea, which is intended to improve the results in Figs. 7 and 8, is inspired from [65,66]. The first snapshot of the new database coincides with the computed θ' field at $t = 280$ s shown in the right panel of Fig. 5 and the total number of snapshots is 155. We use the 85% of the snapshots as training set, following time order for DMD and HDMD and randomly for PODI. We will refer to this subset of the database as the second training set. The choice of taking 85% of the database (i.e., 133 snapshots) is to have the same prediction time for DMD and HDMD as before, i.e., $t = [808, 900]$ s. This will make the comparison with the results in Figs. 7 and 8 fair for DMD/HDMD.

Fig. 9 shows the cumulative energy E in (18) for all the ROMs. We set again the energy threshold δ to 99%, i.e., we retain 49 modes for DMD, 96 modes for HDMD and 52 modes for PODI. Since we are considering only the front propagation phase, the number of modes is less than in the bottom row of Table 3.

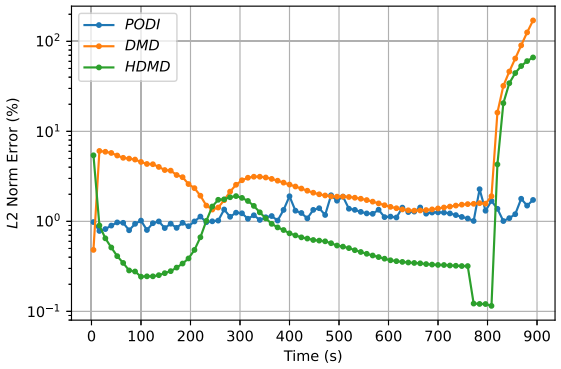

Fig. 8. Density current: time evolution of the L^2 error (37) for DMD, HDMD, and PODI for $\delta = 99\%$.

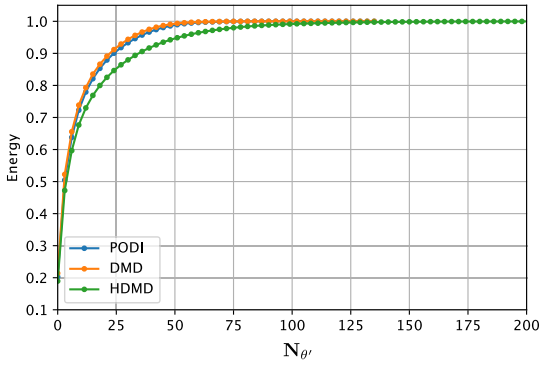

Fig. 9. Density current, second training set: cumulative energy E in (18) for the three ROM approaches under consideration.

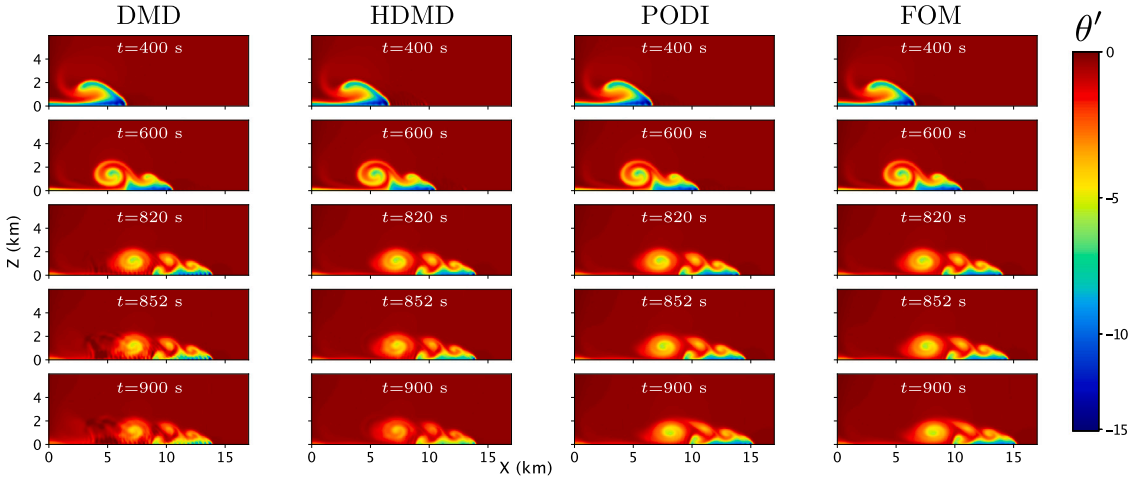

Fig. 10. Density current, second training set: comparison of evolution of θ' given by the ROMs (first 3 columns) and the FOM (last column) for $\delta = 0.99$.

Now that the basis functions have been identified, we proceed with a qualitative comparison reported in Fig. 10. Just like in Fig. 7, the top two rows in Fig. 10 correspond to solutions belonging to the training set. We see an improved system identification for DMD and PODI, with none of the instabilities observed in the first two rows of Fig. 7. By looking at the bottom three rows in Fig. 10, we observe an improvement also in the system prediction, especially in the case of PODI whose solutions are oscillations-free and very similar to the FOM solutions. The DMD solutions, although improved with respect to Fig. 7, are still affected by instabilities for $x < 5$ Km that especially evident for $t = 820$ s and $t = 900$ s.

Finally, let us take a look at the time evolution of L^2 error (37) in Fig. 11. The improvement in accuracy is clearly observable by comparing Fig. 11 with 8. The DMD algorithm shows a significant improvement, with its performance getting very close to the

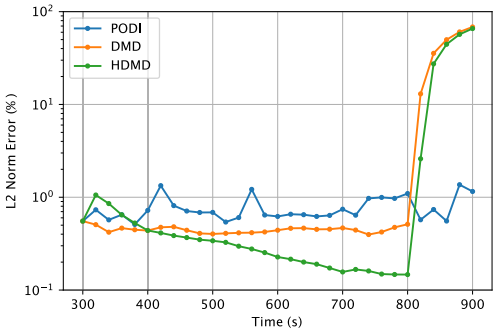

Fig. 11. Density current, second training set: time evolution of the L^2 error (37) for DMD, HDMD, and PODI for energy threshold $\delta = 0.99\%$.

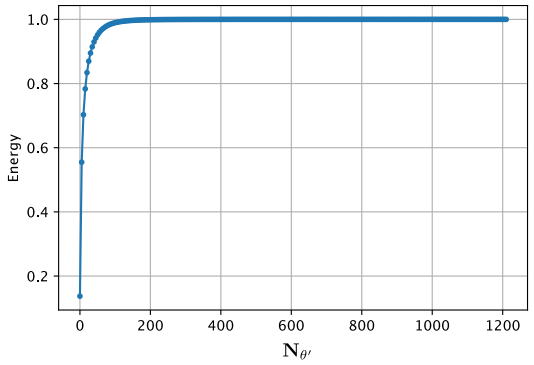

Fig. 12. Density current, parametric study: cumulative energy E in (18) as $N_{\theta'}$ varies.

HDMD. Indeed, the maximum error for DMD decreases from about 120% to about 70%, which is still unacceptable though. The error for PODI remains around 1% throughout the entire time interval. We remark that PODI showed comparable accuracy for the rising thermal bubble when $\delta = 0.99$ (see Fig. 4, bottom panel).

4.2.2. Parametric reconstruction

As mentioned in Section 3, time is the parameter the DMD and HDMD methods were designed to handle, while PODI can be used for computational studies involving physical parameters too. Hence, in this subsection we investigate the accuracy of PODI in a parametric study involving parameter θ_s in (38).

Let $\theta_s \in [5, 10]$ K. Like in the case of time, we choose a uniform sample distribution for θ_s with 11 sampling points. For each of these 11 values of θ_s, we run a simulation for the entire time interval of interest, i.e., $[0, 900]$ s and collect snapshots every 4 s. The total number of snapshots in the database is 2475. Once again, we randomly select 90% of them (i.e., 2227) for the training set. The remaining snapshots are used for validation.

Fig. 12 shows cumulative energy E in (18) as $N_{\theta'}$ varies. We truncated the graph at $N_{\theta'} = 1200$ since E increases very steeply and gets over 99% for rather small values of $N_{\theta'}$. Indeed, to retain 99% of the energy we only need 120 modes.

To evaluate the accuracy of PODI in the parametric study, we consider two values that are not associated to snapshots in the training set: $\theta_s = 6.25$ and $\theta_s = 8.75$. A qualitative comparison between the PODI and FOM solutions is shown in Fig. 13, together with the difference between the two in absolute value. We see that the difference in the FOM and PODI solutions, which does not exceed 1.5 K in absolute value, is localized and confined to the region of the main flow structures. We also observe that the difference in absolute value is smaller during the vertical fall of the bubble (top row in both panels) than during the front propagation (bottom three rows in both panels).

The time evolution of L^2 error (37) for $\theta_s = 6.25$ and $\theta_s = 8.75$ reported in Fig. 14 confirms that the error is smaller initially (i.e., during the fall of the bubble) and increases at later time instances (i.e., when during the front propagation). We remark that for both values of θ_s the error is lower than 10% for the entire time interval. This accuracy can be improved upon by discarding the snapshots associated to the fall of the bubble from the database, as shown in Section 4.2.1.

4.3. 3D rising thermal bubble

The aim of this section is to demonstrate that the applicability of the ROMs under consideration is not limited to problems in 2D. For this purpose, we consider a tree-dimensional variant of the rising thermal bubble benchmark in Section 4.1 [62]. We will restrict the focus to HDMD and PODI, given the poor accuracy of the DMD results presented in the previous two sections.

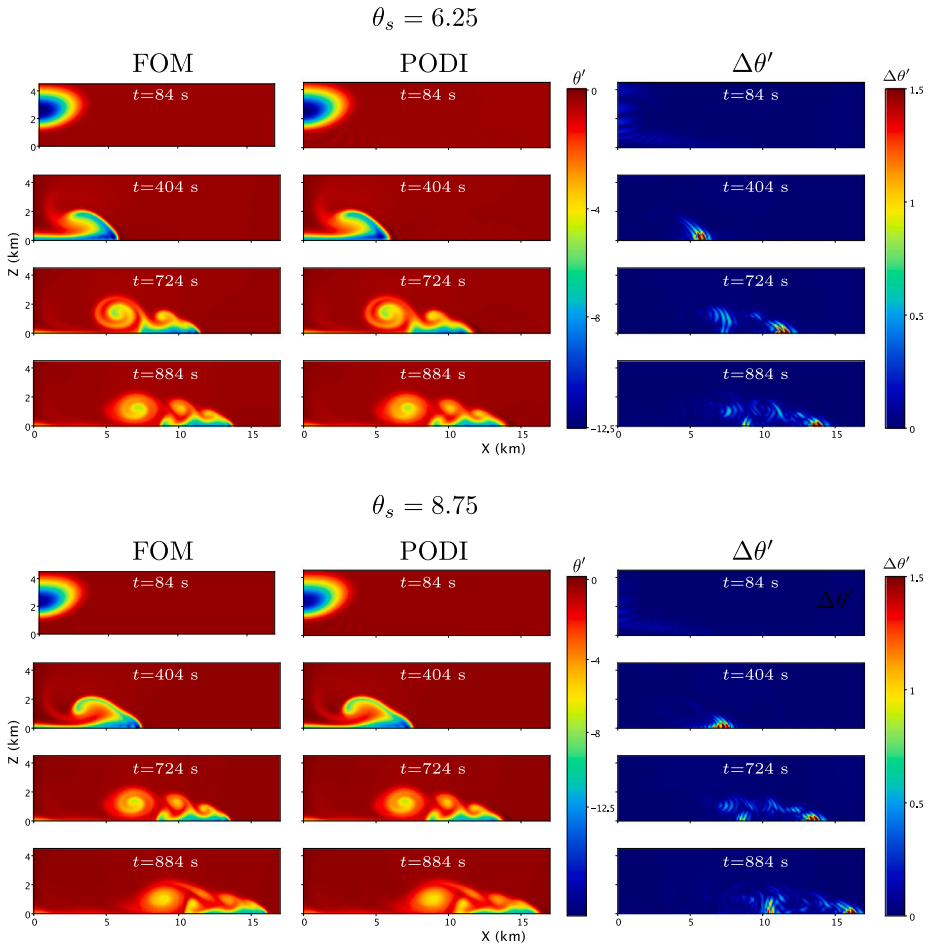

Fig. 13. Density current, parametric study: FOM solutions (left column), PODI solutions (center column), and difference between the two in absolute value (third column) for $\theta_s = 6.25$ (top panel) and $\theta_s = 8.75$ in (bottom panel).

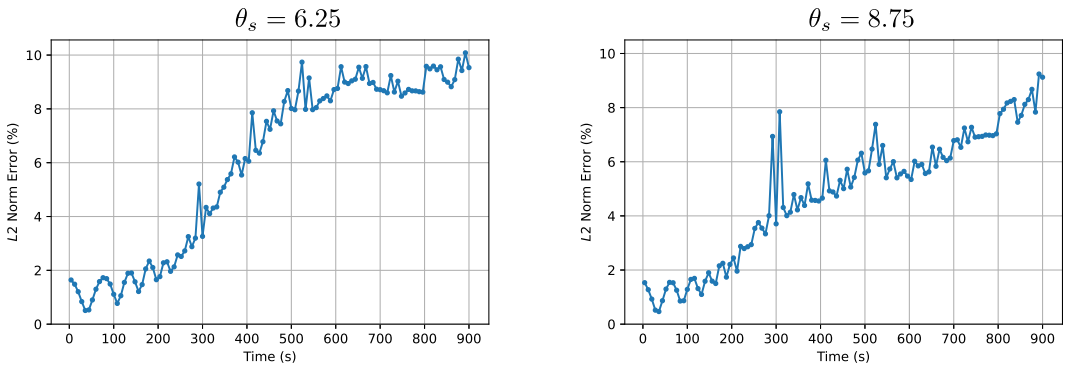

Fig. 14. Density current, parametric study: evolution of the L^2 error (37) for $\theta_s = 6.25$ (left) and $\theta_s = 8.75$ (right).

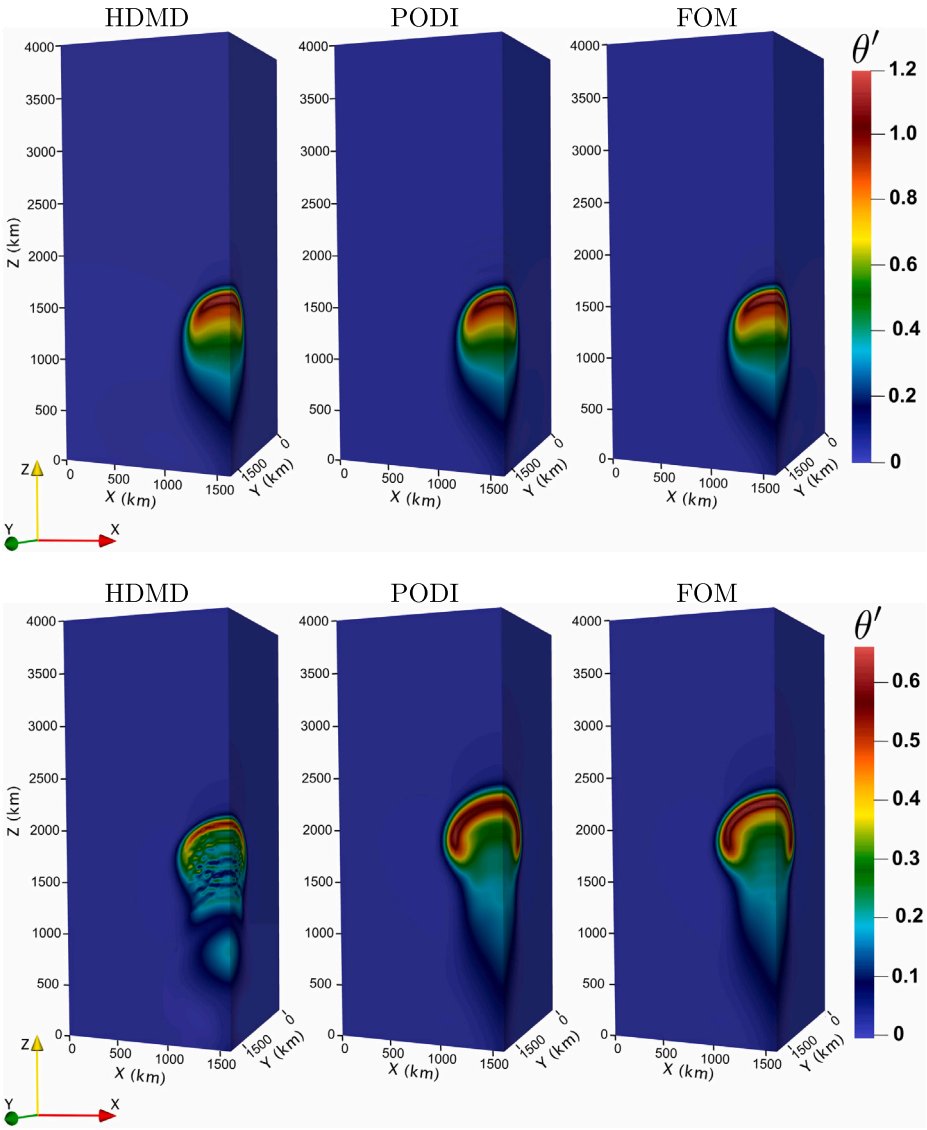

Fig. 15. 3D rising thermal bubble: comparison of θ' computed by HDMD (left), PODI (center), and FOM (right) for $t = 320$ s (top) and $t = 480$ s (bottom). We set $\delta = 0.99$.

The computational domain for this benchmark is $\Omega = [0, 1600] \times [0, 1600] \times [0, 4000]$ m^3 and the time interval of interest is $(0, 500]$ s. We start from a neutrally stratified atmosphere with uniform background potential temperature θ^0=300 K perturbed by a spheric bubble of warmer air. The initial temperature field is

$$\theta^0 = 300 + 2\left[1 - \frac{r}{r_0}\right] \text{ if } r \leq r_0 = 500 \text{ m}, \quad \theta^0 = 300 \text{ otherwise}, \tag{39}$$

where $(x_c, y_c, z_c) = (1600, 1600, 500)$ m is the center and $r = \sqrt{(x - x_c)^2 + (y - y_c)^2 + (z - z_c)^2}$ the radius of the spheric perturbation. The initial density is given by (35), while the initial specific enthalpy is (36). The initial velocity field is zero everywhere. We impose impenetrable, free-slip boundary conditions on all the boundaries.

We generate a uniform structured mesh with mesh size $h = \Delta x = \Delta y = \Delta z = 40$ m and set the time step set to $\Delta t = 0.1$ s. We set $\mu_a = 12.5$ and $Pr = 1$ in (8)–(9). These values allow us to get comparable results to those obtained the variational multiscale method in [62].

To generate the reduced basis, we collect a database of 100 snapshots, i.e., the computed θ' every 5 s. The training set contains 80% of the snapshots (i.e., 80 snapshots) while the validation set contains the remaining 20% (i.e., 20 snapshots). In the case of PODI, the 80 snapshots in the training set are selected randomly from the entire time interval. For HDMD, the training set contains

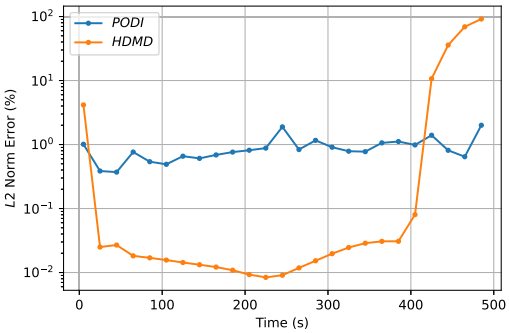

Fig. 16. 3D rising thermal bubble: evolution of the L^2 error (37) for HDMD and PODI with energy threshold $\delta = 0.99\%$.

the first 80 snapshots and we set $M = 40$, which is larger that the value we used for the corresponding 2D benchmark. Notice that we have double the length of the prediction window, which was 10% of the total time interval for the 2D warm bubble.

We set the energy threshold δ to 99%, corresponding to 45 modes for HDMD and 22 modes for PODI. Fig. 15 compares the potential temperature perturbation given by HDMD and PODI with the FOM solution for two time steps: $t = 320$ s (system identification) and $t = 480$ s (prediction). We observe that both HDMD and PODI interpolate well the dynamics of the system. On the other hand, unlike the 2D case, HDMD completely fails at predicting the solution.

To quantify the agreement between the ROM solutions and the FOM solution, we plot in Fig. 16 the time evolution of L^2 error (37). We see that HDMD does a very poor job in predicting the future behavior of the system in the L^2 norm: the magnitude of the error at the end of the time interval is around 96%, about 5 times greater than the 2D case. On the other hand, the error for PODI remains around 1% throughout the entire time interval.

5. Concluding remarks

With the goal of reducing the computational time to forecast regional atmospheric flow, we considered three data driven reduced order modeling techniques: a ROM specifically designed for system prediction called DMD, an improvement of DMD called HDMD, and an interpolatory ROM called PODI. PODI has the advantage over DMD and HDMD to allow for parametric studies, i.e., it can handle physical parameters in the same way it handles time. We applied the three ROMs to two well-known 2D benchmarks for mesoscale flow and compared their accuracy in system identification and prediction of the system behavior, and in terms of computational time. In addition, we present preliminary results in 3D. Since the use of ROMs for the prediction of atmospheric flow is still in its infancy, our work in this paper has a few distinguishing elements: (i) the ROMs are applied to the simulation of mesoscale flow, which features higher resolution than the simulation of global circulation; (ii) the ROMs are used for both system identification and prediction; and (iii) one ROM is used for a parametric study.

In the case where time is the only parameter of interest, our results show that all three ROMs are accurate in the identification of the system dynamics, although local instabilities are seen in the DMD and PODI solutions. The price to pay for the lack of oscillations and increased accuracy in the HDMD solutions is a substantial increase in computational time: the time of a HDMD simulation is two orders of magnitude larger than the time of a DMD or PODI simulation. Although DMD and HDMD are intended for forecasts, the accuracy in the prediction of the system dynamics is low even when 99% of the eigenvalue energy is retained and the snapshots in the database are tailored to the problem at hand. Thanks to the interpolatory approach, PODI maintains a good level of accuracy during the entire time interval of interest. This is true also when a physical parameter is varied within a parametric study and for a 3D test case.

Given the promising results obtained with PODI, interesting future research directions for this method include more complex problems, longer prediction windows, and high-dimensional parameter space. In addition, we believe that the results presented in this paper can be improved upon by using Machine Learning-based techniques that can better detect and reproduce the nonlinear behavior exhibited by the full order model. In particular, Convolutional Autoencoders and Long-Short Time Memory could improve both the accuracy and efficiency of the methods in this paper [67–70].

CRediT authorship contribution statement

Arash Hajisharifi: Software, Analysis of the computational data, Visualization, Conceptualization. **Michele Girfoglio:** Conceptualization, Software, Writing – original draft, Review and editing. **Annalisa Quaini:** Supervision, Writing – original draft, Review and editing. **Gianluigi Rozza:** Supervision, Funding acquisition, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

We acknowledge the support provided by the European Research Council Executive Agency by the Consolidator Grant project AROMA-CFD "Advanced Reduced Order Methods with Applications in Computational Fluid Dynamics" - GA 681447, H2020-ERC CoG 2015 AROMA-CFD, the European Union's Horizon 2020 research and innovation program under the Marie Skłodowska-Curie Actions, grant agreement 872442 (ARIA), PON "Research and Innovation on Green related issues" FSE REACT-EU 2021 project, PRIN NA FROM-PDEs project, European High-Performance Computing Joint Undertaking project Eflows4HPC GA N. 955558 and INdAM-GNCS 2019–2021 projects. This work was also partially supported by US National Science Foundation through grant DMS-1953535 (PI A. Quaini).

References

[1] Benner Peter, Stefano Grivet-Talocia, Quarteroni Alfio, Rozza Gianluigi, Schilders Wil, Luis Miguel Silveira, et al., Model Order Reduction. Volume 1: System-and Data-Driven Methods and Algorithms, De Gruyter, 2021.
[2] Peter Benner, Wil Schilders, Stefano Grivet-Talocia, Alfio Quarteroni, Gianluigi Rozza, Luís Miguel Silveira, Model Order Reduction: Volume 2: Snapshot-Based Methods and Algorithms, De Gruyter, 2020.
[3] Peter Benner, Wil Schilders, Stefano Grivet-Talocia, Alfio Quarteroni, Gianluigi Rozza, Luís Miguel Silveira, Model order reduction: volume 3 applications, De Gruyter, 2020.
[4] Jan S. Hesthaven, Gianluigi Rozza, Benjamin Stamm, Certified Reduced Basis Methods for Parametrized Partial Differential Equations, Vol. 590, Springer, 2016.
[5] Muhammad Haris Malik, Reduced Order Modeling for Smart Grids' Simulation and Optimization (Ph.D. thesis), École centrale de Nantes; Universitat politécnica de Catalunya, 2017.
[6] Gianluigi Rozza, Dinh Bao Phuong Huynh, Anthony T. Patera, Reduced basis approximation and a posteriori error estimation for affinely parametrized elliptic coercive partial differential equations: Application to transport and continuum mechanics, Arch. Comput. Methods Eng. 15 (3) (2008) 229–275.
[7] Maxime Barrault, Yvon Maday, Ngoc Cuong Nguyen, Anthony T Patera, An 'empirical interpolation'method: Application to efficient reduced-basis discretization of partial differential equations, C. R. Math. 339 (9) (2004) 667–672.
[8] Saifon Chaturantabut, Danny C. Sorensen, Nonlinear model reduction via discrete empirical interpolation, SIAM J. Sci. Comput. 32 (5) (2010) 2737–2764.
[9] J Nathan Kutz, Steven L Brunton, Bingni W Brunton, Joshua L Proctor, Dynamic Mode Decomposition: Data-Driven Modeling of Complex Systems, SIAM, 2016.
[10] Peter J. Schmid, Dynamic mode decomposition of numerical and experimental data, J. Fluid Mech. 656 (2010) 5–28.
[11] Peter J. Schmid, Larry Li, Matthew P. Juniper, O. Pust, Applications of the dynamic mode decomposition, Theor. Comput. Fluid Dyn. 25 (2011) 249–259.
[12] Jonathan H Tu, Clarence W Rowley, Dirk M Luchtenburg, Steven L Brunton, J Nathan. Kutz, On dynamic mode decomposition: Theory and applications, J. Comput. Dyn. 1 (2014/12) 391–421.
[13] Peter J. Schmid, Application of the dynamic mode decomposition to experimental data, Exper. fluids 50 (2011) 1123–1130.
[14] Daniel Duke, Damon Honnery, Julio Soria, Experimental investigation of nonlinear instabilities in annular liquid sheets, J. Fluid Mech. 691 (2012) 594–604.
[15] Abu Seena, Hyung Jin Sung, Dynamic mode decomposition of turbulent cavity flows for self-sustained oscillations, Int. J. Heat Fluid Flow 32 (6) (2011) 1098–1110.
[16] Hassan Arbabi, Igor Mezic, Ergodic theory, dynamic mode decomposition, and computation of spectral properties of the Koopman operator, SIAM J. Appl. Dyn. Syst. 16 (4) (2017) 2096–2126.
[17] Christopher W Curtis, D Jay Alford-Lago, Erik Bollt, Andrew Tuma, Machine learning enhanced Hankel Dynamic-Mode Decomposition, Chaos 33 (8) (2023).
[18] Keisuke Fujii, Naoya Takeishi, Benio Kibushi, Motoki Kouzaki, Yoshinobu Kawahara, Data-driven spectral analysis for coordinative structures in periodic human locomotion, Sci. Rep. 9 (1) (2019) 16755.
[19] Huiming Jiang, Jin Chen, Guangming Dong, Tao Liu, Gang Chen, Study on Hankel matrix-based SVD and its application in rolling element bearing fault diagnosis, Mech. Syst. Signal Process. 52 (2015) 338–359.
[20] Enio Vasconcelos Filho, Paulo Lopes dos Santos, A dynamic mode decomposition approach with Hankel blocks to forecast multi-channel temporal series, IEEE Control Syst. Lett. 3 (3) (2019) 739–744.
[21] Deyou Yang, Han Gao, Guowei Cai, Zhe Chen, Linxin Wang, Jin Ma, Dexin Li, Synchronized ambient data-based extraction of interarea modes using Hankel block-enhanced DMD, Int. J. Electr. Power Energy Syst. 128 (2021) 106687.
[22] Peter Frame, Aaron Towne, Space-time POD and the Hankel matrix, Plos one 18 (8) (2023) e0289637.
[23] Martin W. Hess, Annalisa Quaini, Gianluigi Rozza, A data-driven surrogate modeling approach for time-dependent incompressible Navier-Stokes equations with dynamic mode decomposition and manifold interpolation, Adv. Comput. Math. 49 (2) (2023) 22.
[24] Nirav Vasant Shah, Michele Girfoglio, Peregrina Quintela, Gianluigi Rozza, Alejandro Lengomin, Francesco Ballarin, Patricia Barral, Finite element based model order reduction for parametrized one-way coupled steady state linear thermo-mechanical problems, Finite Elem. Anal. Des. 212 (2022) 103837.
[25] Michele Girfoglio, Francesco Ballarin, Giuseppe Infantino, Francesca Nicoló, Andrea Montalto, Gianluigi Rozza, Roberto Scrofani, Marina Comisso, Francesco Musumeci, Non-intrusive PODI-ROM for patient-specific aortic blood flow in presence of a LVAD device, Med. Eng. Phys. 107 (2022) 103849.
[26] Arash Hajisharifi, Francesco Romanò, Michele Girfoglio, Andrea Beccari, Domenico Bonanni, Gianluigi Rozza, A non-intrusive data-driven reduced order model for parametrized CFD-DEM numerical simulations, J. Comput. Phys. 491 (2023) 112355.
[27] Nicola Demo, Marco Tezzele, Andrea Mola, Gianluigi Rozza, An efficient shape parametrisation by free-form deformation enhanced by active subspace for hull hydrodynamic ship design problems in open source environment, in: ISOPE International Ocean and Polar Engineering Conference, ISOPE, 2018, pp. ISOPE–I.
[28] Nicola Demo, Marco Tezzele, Gianluca Gustin, Gianpiero Lavini, Gianluigi Rozza, Shape optimization by means of proper orthogonal decomposition and dynamic mode decomposition, in: Technology and Science for the Ships of the Future: Proceedings of NAV 2018: 19th International Conference on Ship & Maritime Research, 2018, pp. 212–219.
[29] Matteo Ripepi, Mark Johannes Verveld, NW Karcher, Thomas Franz, Mohammad Abu-Zurayk, Stefan Görtz, TM Kier, Reduced-order models for aerodynamic applications, loads and MDO, CEAS Aeronaut. J. 9 (1) (2018) 171–193.

[30] Andrea Lario, Romit Maulik, Oliver T Schmidt, Gianluigi Rozza, Gianmarco Mengaldo, Neural-network learning of SPOD latent dynamics, J. Comput. Phys. 468 (2022) 111475.

[31] Suraj Pawar, Omer San, Equation-free surrogate modeling of geophysical flows at the intersection of machine learning and data assimilation, J. Adv. Modelling Earth Syst. 14 (11) (2022) e2022MS003170.

[32] Oliver T Schmidt, Gianmarco Mengaldo, Gianpaolo Balsamo, Nils P Wedi, Spectral empirical orthogonal function analysis of weather and climate data, Mon. Weather Rev. 147 (8) (2019) 2979–2995.

[33] Jaideep Pathak, Brian Hunt, Michelle Girvan, Zhixin Lu, Edward Ott, Model-free prediction of large spatiotemporally chaotic systems from data: A reservoir computing approach, Phys. Rev. Lett. 120 (2) (2018) 024102.

[34] Stephan Rasp, Nils Thuerey, Data-driven medium-range weather prediction with a resnet pretrained on climate simulations: A new model for weatherbench, J. Adv. Modelling Earth Syst. 13 (2) (2021) e2020MS002405.

[35] Martin G Schultz, Clara Betancourt, Bing Gong, Felix Kleinert, Michael Langguth, Lukas Hubert Leufen, Amirpasha Mozaffari, Scarlet Stadtler, Can deep learning beat numerical weather prediction? Phil. Trans. R. Soc. A 379 (2194) (2021) 20200097.

[36] Jonathan A. Weyn, Dale R. Durran, Rich Caruana, Can machines learn to predict weather? Using deep learning to predict gridded 500-hPa geopotential height from historical weather data, J. Adv. Modelling Earth Syst. 11 (8) (2019) 2680–2693.

[37] Remi Lam, Alvaro Sanchez-Gonzalez, Matthew Willson, Peter Wirnsberger, Meire Fortunato, Alexander Pritzel, Suman Ravuri, Timo Ewalds, Ferran Alet, Zach Eaton-Rosen, et al., GraphCast: Learning skillful medium-range global weather forecasting, 2022, arXiv preprint arXiv:2212.12794.

[38] Jaideep Pathak, Shashank Subramanian, Peter Harrington, Sanjeev Raja, Ashesh Chattopadhyay, Morteza Mardani, Thorsten Kurth, David Hall, Zongyi Li, Kamyar Azizzadenesheli, et al., FourCastNet: A global data-driven high-resolution weather model using adaptive fourier neural operators, 2022, arXiv preprint arXiv:2202.11214.

[39] Kaifeng Bi, Lingxi Xie, Hengheng Zhang, Xin Chen, Xiaotao Gu, Qi Tian, Accurate medium-range global weather forecasting with 3D neural network, Nature 619 (2023) 533–538.

[40] Nicola Clinco, Michele Girfoglio, Annalisa Quaini, Gianluigi Rozza, Filter stabilization for the mildly compressible Euler equations with application to atmosphere dynamics simulations, Comput. Fluids 266 (2023) 106057.

[41] Michele Girfoglio, Annalisa Quaini, Gianluigi Rozza, Validation of an OpenFOAM®-based solver for the Euler equations with benchmarks for mesoscale atmospheric modeling, AIP Adv. 13 (5) (2023).

[42] Simone Marras, Murtazo Nazarov, Francis X. Giraldo, Stabilized high-order Galerkin methods based on a parameter-free dynamic SGS model for LES, J. Comput. Phys. 301 (2015) 77–101.

[43] James F. Kelly, Francis X. Giraldo, Continuous and discontinuous Galerkin methods for a scalable three-dimensional nonhydrostatic atmospheric model: Limited-area mode, J. Comput. Phys. 231 (24) (2012) 7988–8008.

[44] Suhas V. Patankar, D. Brian Spalding, A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows, in: Numerical Prediction of Flow, Heat Transfer, Turbulence and Combustion, Elsevier, 1983, pp. 54–73.

[45] Raad I. Issa, Solution of the implicitly discretised fluid flow equations by operator-splitting, J. Comput. Phys. 62 (1) (1986) 40–65.

[46] Fadl Moukalled, Luca Mangani, Marwan Darwish, F Moukalled, L Mangani, M Darwish, The Finite Volume Method, Springer, 2016.

[47] G.E.A. - Geophysical and Environmental Applications, https://github.com/GEA-Geophysical-and-Environmental-Apps/GEA.

[48] Michele Girfoglio, Annalisa Quaini, Gianluigi Rozza, GEA: A new finite volume-based open source code for the numerical simulation of atmospheric and ocean flows, 2023, https://https://arxiv.org/abs/2303.10499.

[49] Marco Tezzele, Nicola Demo, Mahmoud Gadalla, Andrea Mola, Gianluigi Rozza, Model order reduction by means of active subspaces and dynamic mode decomposition for parametric hull shape design hydrodynamics, in: Technology and Science for the Ships of the Future: Proceedings of NAV 2018: 19th International Conference on Ship and Maritime Research, IOS Press, 2018, pp. 569–576.

[50] Clarence W. Rowley, Igor Mezić, Shervin Bagheri, Philipp Schlatter, Dan S Henningson, Spectral analysis of nonlinear flows, J. Fluid Mech. 641 (2009) 115–127.

[51] Francesco Andreuzzi, Nicola Demo, Gianluigi Rozza, A dynamic mode decomposition extension for the forecasting of parametric dynamical systems, SIAM J. Appl. Dyn. Syst. 22 (3) (2023) 2432–2458.

[52] Joshua L. Proctor, Steven L. Brunton, J. Nathan Kutz, Dynamic mode decomposition with control, SIAM J. Appl. Dyn. Syst. 15 (1) (2016) 142–161.

[53] Nina Golyandina, Vladimir Nekrutkin, Anatoly A. Zhigljavsky, Analysis of Time Series Structure: SSA and Related Techniques, CRC Press, 2001.

[54] Bubathi Muruganatham, MA Sanjith, B Krishnakumar, SAV Satya Murty, Roller element bearing fault diagnosis using singular spectrum analysis, Mech. Syst. Signal Process. 35 (1–2) (2013) 150–166.

[55] T. Bui-Thanh, Murali Damodaran, Karen Willcox, Proper orthogonal decomposition extensions for parametric applications in compressible aerodynamics, in: 21st AIAA Applied Aerodynamics Conference, 2003, p. 4213.

[56] Martin D. Buhmann, Radial BAsis FUnctions: THeory and IMplementations (Cambridge Monographs on Applied and Computational Mathematics; 12), Cambridge University Press, 2003.

[57] Nashat N. Ahmad, John Lindeman, Euler solutions using flux-based wave decomposition, Internat. J. Numer. Methods Fluids 54 (1) (2007) 47–72.

[58] Nashat N. Ahmad, High-resolution wave propagation method for stratified flows, in: 2018 Atmospheric and Space Environments Conference, 2018, p. 3498.

[59] Yongliang Feng, Johann Miranda-Fuentes, Jérôme Jacob, Pierre Sagaut, Hybrid lattice Boltzmann model for atmospheric flows under anelastic approximation, Phys. Fluids 33 (3) (2021) 036607.

[60] Richard L Carpenter Jr., Kelvin K Droegemeier, Paul R Woodward, Carl E Hane, Application of the piecewise parabolic method (PPM) to meteorological modeling, Mon. Weather Rev. 118 (3) (1990) 586–612.

[61] Francis X. Giraldo, Marco Restelli, A study of spectral element and discontinuous Galerkin methods for the Navier–Stokes equations in nonhydrostatic mesoscale atmospheric modeling: Equation sets and test cases, J. Comput. Phys. 227 (8) (2008) 3849–3877.

[62] Simone Marras, Margarida Moragues, Mariano Vázquez, Oriol Jorba, Guillaume Houzeaux, A variational multiscale stabilized finite element method for the solution of the Euler equations of nonhydrostatic stratified flows, J. Comput. Phys. 236 (2013) 380–407.

[63] Jerry M Straka, Robert B Wilhelmson, Louis J Wicker, John R Anderson, Kelvin K Droegemeier, Numerical solutions of a non-linear density current: A benchmark solution and comparisons, Internat. J. Numer. Methods Fluids 17 (1) (1993) 1–22.

[64] N. Demo, M. Tezzele, G. Rozza, EZyRB: Easy reduced basis method, J. Open Source Softw. 3 (24) (2018) 661.

[65] Michele Girfoglio, Annalisa Quaini, Gianluigi Rozza, A hybrid projection/data-driven reduced order model for the Navier-Stokes equations with nonlinear filtering stabilization, J. Comput. Phys. 486 (2023) 112127.

[66] Maria Strazzullo, Michele Girfoglio, Francesco Ballarin, Traian Iliescu, Gianluigi Rozza, Consistency of the full and reduced order models for evolve-filter-relax regularization of convection-dominated, marginally-resolved flows, Internat. J. Numer. Methods Engrg. 123 (14) (2022) 3148–3178.

[67] Francisco J. Gonzalez, Maciej Balajewicz, Deep convolutional recurrent autoencoders for learning low-dimensional feature dynamics of fluid systems, 2018, arXiv preprint arXiv:1808.01346.

[68] Romit Maulik, Bethany Lusch, Prasanna Balaprakash, Reduced-order modeling of advection-dominated systems with recurrent neural networks and convolutional autoencoders, Phys. Fluids 33 (3) (2021).

[69] Arvind Mohan, Don Daniel, Michael Chertkov, Daniel Livescu, Compressed convolutional LSTM: An efficient deep learning framework to model high fidelity 3D turbulence, 2019, arXiv preprint arXiv:1903.00033.

[70] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, Wang-chun Woo, Convolutional LSTM network: A machine learning approach for precipitation nowcasting, Adv. Neural Inform. Process. Syst. 28 (2015).