J. Fluid Mech. (2019), *vol.* 872, *pp.* 963–994.
doi:10.1017/jfm.2019.358

Construction of reduced-order models for fluid flows using deep feedforward neural networks

Hugo F. S. Lui[1] **and William R. Wolf**[1],†

[1]School of Mechanical Engineering, Universidade Estadual de Campinas, Campinas, SP 13083-860, Brazil

(Received 4 October 2018; revised 16 April 2019; accepted 27 April 2019; first published online 14 June 2019)

We present a numerical methodology for construction of reduced-order models (ROMs) of fluid flows through the combination of flow modal decomposition and regression analysis. Spectral proper orthogonal decomposition is applied to reduce the dimensionality of the model and, at the same time, filter the proper orthogonal decomposition temporal modes. The regression step is performed by a deep feedforward neural network (DNN), and the current framework is implemented in a context similar to the sparse identification of nonlinear dynamics algorithm. A discussion on the optimization of the DNN hyperparameters is provided for obtaining the best ROMs and an assessment of these models is presented for a canonical nonlinear oscillator and the compressible flow past a cylinder. Then the method is tested on the reconstruction of a turbulent flow computed by a large eddy simulation of a plunging airfoil under dynamic stall. The reduced-order model is able to capture the dynamics of the leading edge stall vortex and the subsequent trailing edge vortex. For the cases analysed, the numerical framework allows the prediction of the flow field beyond the training window using larger time increments than those employed by the full-order model. We also demonstrate the robustness of the current ROMs constructed via DNNs through a comparison with sparse regression. The DNN approach is able to learn transient features of the flow and presents more accurate and stable long-term predictions compared to sparse regression.

Key words: computational methods, low-dimensional models

1. Introduction

Current supercomputers allow the application of high fidelity numerical simulations of turbulent flows over large-scale industrial configurations. The results from these simulations certainly improve the understanding of complex physical phenomena such as mixing enhancement, drag reduction, heat transfer and noise generation, to name a few. Such simulations may lead to discretizations with billions of degrees of freedom in order to resolve the energetically relevant spatial and temporal scales. In these cases, the fluid flow data is generally obtained for long periods using small time steps to compute meaningful converged statistics of the flow with sufficient accuracy.

The analysis of unsteady flows by time-resolved simulations and experiments require the acquisition and treatment of large datasets. In recent years, data-driven algorithms have been developed and applied to perform statistical post-processing of such datasets

† Email address for correspondence: wolf@fem.unicamp.br

of unsteady fluid flows allowing the investigation of complex physical mechanisms in turbulent flows and improving their analyses. For example, one can cite techniques of flow modal decomposition such as proper orthogonal decomposition (POD) and its variations (Lumley 1967; Sieber, Paschereit & Oberleithner 2016; Ribeiro & Wolf 2017; Towne, Schmidt & Colonius 2018), dynamic mode decomposition (DMD) and variations (Schmid 2010; Tu *et al.* 2014; Clainche & Vega 2017), Lagrangian coherent structures (LCS) (Green, Rowley & Haller 2007; Haller 2015; Rockwood, Taira & Green 2016) and resolvent analysis (Luhar, Sharma & McKeon 2014; Sharma *et al.* 2016), among others. Recently, Taira *et al.* (2017) published a review of such techniques in the context of fluid flows.

The previous techniques of modal analysis can also be employed for the construction of reduced-order models (ROMs) which are appealing since they can be used in the preliminary stages of design due to their inherent reduction in the computational costs compared to large-scale simulations. Such techniques should also be useful in the context of optimization analyses and studies of flow control (Brunton & Noack 2015). However, in order to be employed for these applications, ROMs should be able to reproduce the main physical aspects of the full scale models. Several ROM techniques have been discussed in the literature such as Galerkin projection (Rowley, Colonius & Murray 2004), least-squares Petrov–Galerkin projection (Carlberg, Bou-Mosleh & Farhat 2011), eigensystem realization analysis (Juang & Pappa 1985) and sparse regression of nonlinear dynamics (Brunton, Proctor & Kutz 2016). The previous techniques can be used to reduce the original sets of partial differential equations to sets of ordinary differential equations. Sparse regression has also been applied for discovering sets of partial differential equations through spatio-temporal data collection (Rudy *et al.* 2017). These techniques have mostly been applied for canonical problems and their application to more complex turbulent flows is still a challenging task.

In some cases, ROMs constructed using some of the above techniques may exhibit unstable behaviour (Carlberg *et al.* 2011). For example, ROMs developed based on Galerkin projection employ POD to rewrite the Navier–Stokes equations as sets of dynamical systems for the evolution of the POD temporal modes. Due to the basis truncation typical of such methods, numerical instabilities can be created from the imbalance in the turbulent kinetic energy budget in the ROM. This issue may be addressed using turbulence models (Cazemier, Verstappen & Veldman 1998; Östh *et al.* 2014; Protas, Noack & Östh 2015), however, this approach can destroy consistency between the original partial differential equations and the ODE system of the ROM. Other alternatives have been also proposed to deal with this problem via minimal rotation of the projected subspace (Balajewicz, Tezaur & Dowell 2016). This approach is able to account for the contribution of the truncated modes while keeping the consistency between the full- and reduced-order models. Recently, San & Maulik (2018) employed machine learning via neural networks to compute optimal coefficients for an eddy viscosity model which is able to stabilize a POD–Galerkin ROM.

Machine learning is a field that has applications in several areas from data classification to pattern recognition and nonlinear function approximation. Several groups have applied machine learning for investigations concerning fluid flows. Ling & Templeton (2015) evaluated different machine learning techniques to classify regions of high uncertainty in RANS calculations. Ling, Kurzawski & Templeton (2016) presented a novel deep neural network (DNN) architecture to improve RANS turbulence models. In a similar context, Wang, Wu & Xiao (2017) developed a

machine learning approach based on random forests for predicting discrepancies in Reynolds stresses obtained by RANS calculations. Ströfer *et al.* (2019) employed convolutional neural networks, a machine learning technique well suited for image pattern recognition, to identify features in fluid flows.

Machine learning is also a natural candidate for the development of ROMs of large-scale dynamical systems, typical of numerical simulations of unsteady flows. A discussion of the application of deep learning in the context just described was presented by Kutz (2017). Recently, several authors have proposed algorithms for the prediction of high-dimensional complex dynamical systems using neural networks and their variants (Pan & Duraisamy 2018; Rudy, Kutz & Brunton 2018; San & Maulik 2018; Vlachas *et al.* 2018; Wan *et al.* 2018). Rudy *et al.* (2018) presented a methodology that is able to learn the dynamics of a particular system and estimate the noise from measurements using feedforward neural networks (FNN). Pan & Duraisamy (2018) performed a comparison between FNNs and sparse regression for modelling of dynamical systems and demonstrated the benefits of the former in terms of adaptability. They computed the Frobenius norm of the Jacobian of the neural network as the regularization term of the cost function to improve the accuracy and robustness of a framework. Wan *et al.* (2018) developed a ROM methodology capable of modelling extreme events occurring in dynamical systems. These authors employed a long short-term memory (LSTM) approach to regularize a recurrent neural network which obtains the complementary dynamics of a nonlinear Galerkin projection of the dynamical system. Vlachas *et al.* (2018) combined a LSTM neural network with a mean stochastic model to propose a data-driven algorithm which has desirable short- and long-term prediction capabilities.

In this work we present a numerical methodology which combines flow modal decomposition via POD and regression analysis using machine learning and FNNs. The framework is implemented in a context similar to that of the sparse identification of nonlinear dynamics (SINDy) algorithm (Brunton *et al.* 2016). In order to improve the regression step in the current approach, a spectral proper orthogonal decomposition (SPOD) (Sieber *et al.* 2016) is employed to extract a low-dimensional representation of the full-order model (FOM). The SPOD technique is able to filter the temporal modes while preserving the information of the FOM by a redistribution of the energy of the system to higher POD modes. The current method is applied for the construction of ROMs of unsteady compressible flows. First, we test the method for a simple two degree of freedom nonlinear dynamical system. Then, we evaluate the capability of the method to reproduce the compressible flow past a cylinder including its noise generation. In this case, we show that the current DNN approach is also able to reproduce the transient stages of the flow. Finally, the method is tested for a turbulent flow involving dynamic stall of a plunging airfoil. We present a discussion of the optimization of hyperparameters for obtaining the best models for the deep neural networks. For both the cylinder and plunging airfoil cases a comparison between DNN and sparse regression techniques is presented in terms of their predictive capability. The approach presented in this work allows us to predict the flow field beyond the training window and with larger time increments than those used by the FOM, demonstrating the robustness of the current ROMs constructed via DNNs. The current numerical tool for the construction of ROMs via DNNs can be downloaded from http://cces.unicamp.br/software/.

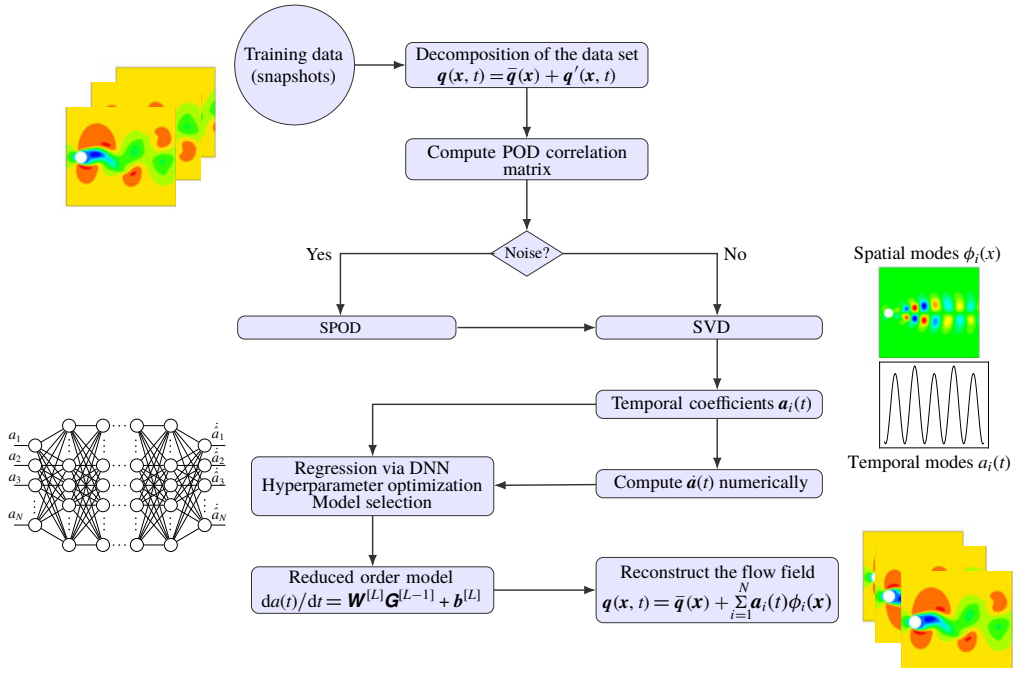

FIGURE 1. (Colour online) Schematic of the current method used for construction of ROMs via DNNs and flow modal decomposition.

2. Theoretical and numerical formulation

2.1. *Flow modal decomposition*

The equations governing an unsteady three-dimensional compressible flow contain partial derivatives with respect to both the spatial coordinates $\boldsymbol{x} = \begin{bmatrix} x & y & z \end{bmatrix}^{\mathrm{T}}$ and time t. Using the method of lines one can first approximate the spatial derivatives producing a system of nonlinear ordinary differential equations (ODEs). In the most general notation, for each mesh point, these ODEs would be expressed in the form

$$\frac{\mathrm{d}\boldsymbol{q}}{\mathrm{d}t} = \boldsymbol{F}(\boldsymbol{q}, t), \tag{2.1}$$

where $\boldsymbol{F}$ is a nonlinear operator, $\boldsymbol{q} = \begin{bmatrix} \rho & \rho u & \rho v & \rho w & p \end{bmatrix}^{\mathrm{T}}$ is the vector of flow variables, ρ is the density, u, v and w are respectively the x-, y- and z-components of the velocity vector and p is the pressure.

Here, we consider that a dynamical system given by (2.1) is solved at each mesh point. To determine the nonlinear operator $\boldsymbol{F}$ from data, we follow the ideas of the SINDy algorithm developed by Brunton *et al.* (2016) with some modifications. A schematic of the current method can be seen in figure 1. First, we collect snapshots of the variables which will be our training data. The data set is then arranged into a matrix $\boldsymbol{Q}$ as

$$\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{q}(\boldsymbol{x}_1, t_1) & \boldsymbol{q}(\boldsymbol{x}_2, t_1) & \cdots & \boldsymbol{q}(\boldsymbol{x}_{N_p}, t_1) \\ \boldsymbol{q}(\boldsymbol{x}_1, t_2) & \boldsymbol{q}(\boldsymbol{x}_2, t_2) & \cdots & \boldsymbol{q}(\boldsymbol{x}_{N_p}, t_2) \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{q}(\boldsymbol{x}_1, t_{N_T}) & \boldsymbol{q}(\boldsymbol{x}_2, t_{N_T}) & \cdots & \boldsymbol{q}(\boldsymbol{x}_{N_p}, t_{N_T}) \end{bmatrix}, \tag{2.2}$$

where N_p is the number of grid points in the computational domain and N_T is the number of snapshots.

Because of the high dimensionality of the input data $\boldsymbol{Q}$, we first reduce the dimension of the dynamical system using the snapshot POD method (Sirovich 1986). Proper orthogonal decomposition has been widely applied to obtain optimal bases that represent the most energetic content of the system dynamics with as few basis functions as possible (Lumley 1967). The snapshot POD approach starts with a decomposition of the vector quantities $\boldsymbol{q}(\boldsymbol{x}, t)$ into the mean flow, $\overline{\boldsymbol{q}}(\boldsymbol{x})$, and fluctuations, $\boldsymbol{q}'(\boldsymbol{x}, t)$. The latter can be further expanded into a combination of spatial modes $\boldsymbol{\phi}_i(\boldsymbol{x})$ and their temporal coefficients $\boldsymbol{a}_i(t)$ for a defined number of N modes as

$$\boldsymbol{q}(\boldsymbol{x}, t) = \overline{\boldsymbol{q}}(\boldsymbol{x}) + \boldsymbol{q}'(\boldsymbol{x}, t) = \overline{\boldsymbol{q}}(\boldsymbol{x}) + \sum_{i=1}^{N} \boldsymbol{a}_i(t)\boldsymbol{\phi}_i(\boldsymbol{x}). \tag{2.3}$$

To calculate the POD correlation matrix of the data set $\boldsymbol{Q}$ some specific norm must be used. For an incompressible flow, a kinetic energy norm provides an optimal result, however, for a compressible flow, other norms can be employed (Rowley *et al.* 2004). Hence, we define a vector $\boldsymbol{\eta} = \begin{bmatrix}\eta_1 & \eta_2 & \eta_3 & \eta_4 & \eta_5\end{bmatrix}^{\mathrm{T}}$ that determines which norm should be used to compute the correlation between two snapshots. For example, a pressure-based norm uses $\boldsymbol{\eta} = \begin{bmatrix}0 & 0 & 0 & 0 & 1\end{bmatrix}^{\mathrm{T}}$ and a kinetic energy norm uses $\boldsymbol{\eta} = \begin{bmatrix}0 & 1 & 1 & 1 & 0\end{bmatrix}^{\mathrm{T}}$. The correlation between two snapshots is computed using the L^2 inner product. Therefore, considering $\boldsymbol{q}'(\boldsymbol{x}, t_i) = \boldsymbol{q}'_i$ and $\boldsymbol{q}'(\boldsymbol{x}, t_j) = \boldsymbol{q}'_j$, the elements of the correlation matrix $\boldsymbol{C}$ are given by

$$C_{ij} = \int_{\Omega} [\eta_1 \rho'_i \rho'_j + \eta_2 (\rho u)'_i (\rho u)'_j + \eta_3 (\rho v)'_i (\rho v)'_j + \eta_4 (\rho w)'_i (\rho w)'_j + \eta_5 p'_i p'_j] \, \mathrm{d}\Omega, \tag{2.4}$$

where Ω is the fluid region of interest for the reconstruction and the matrix $\boldsymbol{C}$ is of size $N \times N$. For the problems studied in this work we employ norms based on kinetic energy and pressure. Despite the changes in the POD modes computed for the different norms, we observed that the ROMs obtained by either kinetic energy or pressure norms were similar. In both cases, stable and accurate models could be reconstructed by the DNNs and further comments are provided in the results section.

In turbulent flows the POD temporal modes may contain contributions from several frequencies including high-frequency noise. For such cases in order to provide a better identification of the individual modes and to smooth out the temporal coefficients, we employ the SPOD described by Sieber *et al.* (2016). The SPOD technique is able to filter the temporal modes while preserving the information of the FOM by a redistribution of the energy of the system to higher POD modes. The technique consists of applying a filter function to the POD correlation matrix which results in a matrix $\tilde{\boldsymbol{C}}$ with elements given as

$$\tilde{C}_{ij} = \sum_{k=-N_f/2}^{N_f/2} g_k C_{i+k,j+k}. \tag{2.5}$$

Here, g_k is the filter function and N_f is the size of the filter window. We consider a periodic temporal series and assume that the correlation matrix is also periodic (see Ribeiro & Wolf (2017) for details). Hence, $N_f = f_{snap}/N_T$, where f_{snap} is the number of snapshots used in the SPOD filter. Following this notation, if we apply 50 % of filter

to the correlation matrix, it means that we are filtering 50 % of the total number of snapshots. Several filter functions can be applied to the POD correlation matrix and their effects are reported by Ribeiro & Wolf (2017). The filter function employed in this work is the box filter represented by $g_k = 1/(2N_f + 1)$.

The temporal coefficients $\boldsymbol{a}_i = \begin{bmatrix} a_i(t_1) & \cdots & a_i(t_{N_T}) \end{bmatrix}^{\mathrm{T}}$ and the POD eigenvalues λ_i are determined from the filtered correlation matrix $\tilde{\boldsymbol{C}}$ as

$$\tilde{\boldsymbol{C}}\boldsymbol{a}_i = \lambda_i \boldsymbol{a}_i. \tag{2.6}$$

Singular value decomposition (SVD) can be employed to compute the eigenvalues λ_i and eigenvectors $\boldsymbol{a}_i$ of $\tilde{\boldsymbol{C}}$ since the matrix is real symmetric positive-definite. The spatial modes are obtained from the projection of the fluctuation quantities onto the temporal coefficients

$$\boldsymbol{\phi}_i(\boldsymbol{x}) = \frac{1}{\lambda_i} \sum_{j=1}^{N} \boldsymbol{a}_i(t_j) \boldsymbol{q}'(\boldsymbol{x}, t_j). \tag{2.7}$$

Finally, the temporal coefficients and spatial modes can be stored in matrices $\boldsymbol{A}$ and $\boldsymbol{\Phi}$ as

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{a}^{\mathrm{T}}(t_1) \\ \boldsymbol{a}^{\mathrm{T}}(t_2) \\ \vdots \\ \boldsymbol{a}^{\mathrm{T}}(t_{N_T}) \end{bmatrix} = \begin{bmatrix} a_1(t_1) & a_2(t_1) & \cdots & a_N(t_1) \\ a_1(t_2) & a_2(t_2) & \cdots & a_N(t_2) \\ \vdots & \vdots & \ddots & \vdots \\ a_1(t_{N_T}) & a_2(t_{N_T}) & \cdots & a_N(t_{N_T}) \end{bmatrix}, \tag{2.8}$$

and

$$\boldsymbol{\Phi} = \begin{bmatrix} \phi_1(\boldsymbol{x_1}) & \phi_1(\boldsymbol{x_2}) & \cdots & \phi_1(\boldsymbol{x}_{N_p}) \\ \phi_2(\boldsymbol{x_1}) & \phi_2(\boldsymbol{x_2}) & \cdots & \phi_2(\boldsymbol{x}_{N_p}) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_N(\boldsymbol{x_1}) & \phi_N(\boldsymbol{x_2}) & \cdots & \phi_N(\boldsymbol{x}_{N_p}) \end{bmatrix}, \tag{2.9}$$

where the temporal coefficients are the columns of $\boldsymbol{A}$ and the spatial modes are the rows of $\boldsymbol{\Phi}$. Hence, the matrix of fluctuation quantities can be written as

$$\boldsymbol{Q}' = \boldsymbol{A}\boldsymbol{\Phi}. \tag{2.10}$$

Taking the time derivative of (2.3), we arrive at the following set of equations

$$\frac{\mathrm{d}\boldsymbol{q}(t)}{\mathrm{d}t} = \sum_{i=1}^{N} \boldsymbol{\phi}_i(\boldsymbol{x}) \frac{\mathrm{d}\boldsymbol{a}_i(t)}{\mathrm{d}t}. \tag{2.11}$$

The last term of (2.11) represents the temporal evolution of coefficients $\boldsymbol{a}_i(t)$ associated with the N modes retained in the SPOD modal basis. We can express this system of coupled ODEs as

$$\frac{\mathrm{d}\boldsymbol{a}(t)}{\mathrm{d}t} = \dot{\boldsymbol{a}}(t) = \boldsymbol{F}(\boldsymbol{a}(t)). \tag{2.12}$$

Next, we compute the derivative $\dot{\boldsymbol{a}}(t)$ numerically using the data $\boldsymbol{a}(t)$ for each temporal mode. Different numerical schemes are tested for the computation of the temporal derivatives and a discussion of the application of explicit schemes and compact schemes will be provided in the results section. For now, let us consider

that the numerical scheme employed for the temporal derivatives is a 10th-order accurate compact scheme (Lele 1992) which provides high spectral resolution being non-dissipative and low-dispersive. The derivative $\dot{\boldsymbol{a}}(t)$ is then obtained as

$$\delta_1\dot{\boldsymbol{a}}_{i-2}+\delta_2\dot{\boldsymbol{a}}_{i-1}+\dot{\boldsymbol{a}}_i+\delta_2\dot{\boldsymbol{a}}_{i+1}+\delta_1\dot{\boldsymbol{a}}_{i+2}=\delta_3\frac{\boldsymbol{a}_{i+3}-\boldsymbol{a}_{i-3}}{6h}+\delta_4\frac{\boldsymbol{a}_{i+2}-\boldsymbol{a}_{i-2}}{4h}+\delta_5\frac{\boldsymbol{a}_{i+1}-\boldsymbol{a}_{i-1}}{2h}. \tag{2.13}$$

In the equation above, h is the time step and the coefficients of the numerical scheme are set as $\delta_1 = 1/20$, $\delta_2 = 1/2$, $\delta_3 = 1/100$, $\delta_4 = 101/150$ and $\delta_5 = 17/12$ for a 10th-order discretization. The system of (2.13) written for each temporal mode can be solved as a pentadiagonal linear system for the unknown derivatives $\dot{\boldsymbol{a}}(t)$. The boundary data points can be computed from the interior points following Olson (2012).

The derivatives $\dot{\boldsymbol{a}}(t)$ are then arranged into a matrix $\dot{\boldsymbol{A}}$

$$\dot{\boldsymbol{A}}=\begin{bmatrix}\dot{\boldsymbol{a}}^{\mathrm{T}}(t_1)\\ \dot{\boldsymbol{a}}^{\mathrm{T}}(t_2)\\ \vdots\\ \dot{\boldsymbol{a}}^{\mathrm{T}}(t_{N_T})\end{bmatrix}=\begin{bmatrix}\dot{a}_1(t_1) & \dot{a}_2(t_1) & \cdots & \dot{a}_N(t_1)\\ \dot{a}_1(t_2) & \dot{a}_2(t_2) & \cdots & \dot{a}_N(t_2)\\ \vdots & \vdots & \ddots & \vdots\\ \dot{a}_1(t_{N_T}) & \dot{a}_2(t_{N_T}) & \cdots & \dot{a}_N(t_{N_T})\end{bmatrix}. \tag{2.14}$$

2.2. *Regression via DNN*

Once the matrix of temporal derivatives is computed, we can set up a regression problem to find the weights $\boldsymbol{W}$ and biases $\boldsymbol{b}$ that determine the function $F(\boldsymbol{a}(t))$ presented in (2.12),

$$\dot{\boldsymbol{A}}=\boldsymbol{W}\boldsymbol{\Theta}(\boldsymbol{A})+\boldsymbol{b}, \tag{2.15}$$

where $\boldsymbol{\Theta}(\boldsymbol{A})$ is the matrix of features. In the SINDy algorithm, Brunton *et al.* (2016) suggest that $\boldsymbol{\Theta}(\boldsymbol{A})$ may consist of constant, polynomial, exponential and trigonometric functions. However, in many cases it is difficult to know what set of features should be extracted from the data. Hence, we use machine learning to circumvent the problem of finding the functions which represent the dynamics of the problem. Therefore, the methodology can discover not only the weights $\boldsymbol{W}$ and biases $\boldsymbol{b}$ but also the features $\boldsymbol{\Theta}(\boldsymbol{A})$. The 'learned' features often result in a better performance when compared to those obtained using 'engineered' features. A learning algorithm can find a proper set of features in minutes or hours, depending on the task complexity. On the other hand, manually engineered features would require a great amount of human time and effort for complex tasks (Goodfellow, Bengio & Courville 2016).

Deep learning methods are feature learning algorithms that can find a proper set of features using multiple layers, from higher layer features defined in terms of lower layer features. Automatically learning features at multiple processing layers allows the learning of complex functions through mapping the input to the output directly from a given data (Bengio 2009). In the present work, a DNN is used to learn the weights $\boldsymbol{W}$, biases $\boldsymbol{b}$ and features $\boldsymbol{\Theta}(\boldsymbol{A})$ of the dynamical systems investigated. Figure 2 shows a sample DNN architecture where the input $\boldsymbol{X}$ of the DNN is the matrix $\boldsymbol{A}$ and the target $\boldsymbol{Y}$ is the matrix $\dot{\boldsymbol{A}}$. The DNN calculation procedure is presented in the following algorithm form 1. In the current work, the open source machine learning framework Tensorflow (Abadi *et al.* 2015) is used for training the DNN.

Once we have learned the parameters $\boldsymbol{W}^{[l]}$ and $\boldsymbol{b}^{[l]}$ of our model (2.15), we can use them to predict the temporal coefficients a_i given the initial conditions $\boldsymbol{a}(t_1)$. The

Algorithm 1: Deep feedforward network (DNN).

Input : Training data $\boldsymbol{X}$, Target $\boldsymbol{Y}$, network depth L, activation function σ, number of hidden units $n^{[l]}$ for layer l, learning rate α, regularization parameter λ, maximum number of iterations n_{iter}, exponential decay rate for the 1st moment estimates β_1, exponential decay rate for the 2nd moment estimates β_2, and small constant for numerical stability ϵ

Output : $\boldsymbol{W}^{[l]}$, $\boldsymbol{b}^{[l]}$, $\hat{\boldsymbol{Y}}$

Initialization of parameters: All the weights $\boldsymbol{W}^{[l]}$ are initialized using Xavier's initialization (Xavier & Yoshua 2010). The biases $\boldsymbol{b}^{[l]}$ are initialized to zero. The matrix $\boldsymbol{W}^{[l]}$ is of size $n^{[l+1]} \times n^{[l]}$ and the vector $\boldsymbol{b}^{[l]}$ is of size $n^{[l]} \times 1$;

Initialization of Adam parameters: Adam parameters $\boldsymbol{V}_{dW^{[l]}}$, $\boldsymbol{V}_{db^{[l]}}$, $\boldsymbol{S}_{dW^{[l]}}$ and $\boldsymbol{S}_{db^{[l]}}$ are initialized to zero. They have the same dimensions as $\boldsymbol{W}^{[l]}$ and $\boldsymbol{b}^{[l]}$;

for $iter = 1$ *to* n_{iter} **do**

  Forward propagation ;

  $\boldsymbol{G}^{[0]} = \boldsymbol{X}$;

  for $l = 1$ *to* L **do**

    $\boldsymbol{Z}^{[l]} = \boldsymbol{W}^{[l]}\boldsymbol{G}^{[l-1]} + \boldsymbol{b}^{[l]}$;

    $\boldsymbol{G}^{[l]} = \sigma(\boldsymbol{Z}^{[l]})$; ▷ `for the last layer` L: $\boldsymbol{G}^{[L]} = \boldsymbol{Z}^{[L]}$

  end

  $\hat{\boldsymbol{Y}} = \boldsymbol{Z}^{[L]}$;

  Cost function: $J = \frac{1}{2N_T}\left[\sum_{j=1}^{N}\sum_{i=1}^{N_T}(\hat{Y}_{i,j} - Y_{i,j})^2 + \lambda\sum_{l=1}^{L}\sum_{k=1}^{n^{[l]}}\sum_{j=1}^{n^{[l+1]}}(W_{j,k}^{[l]})^2\right]$;

  Backward propagation ;

  $\mathrm{d}\boldsymbol{G}^{[L]} = \mathrm{d}\boldsymbol{Z}^{[L]}$; ▷ `as` $\boldsymbol{G}^{[L]} = \boldsymbol{Z}^{[L]}$ `then` $\mathrm{d}\boldsymbol{Z}^{[L]} = \hat{\boldsymbol{Y}} - \boldsymbol{Y}$

  $\mathrm{d}\boldsymbol{W}^{[L]} = \frac{1}{N_T}\left[\mathrm{d}\boldsymbol{Z}^{[L]} \cdot (\boldsymbol{G}^{[L-1]})^{\mathrm{T}}\right]$;

  $\mathrm{d}\boldsymbol{b}^{[L]} = \frac{1}{N_T}\sum_{i=1}^{N_T}\mathrm{d}\boldsymbol{Z}^{[L]}$;

  for $l = L-1$ *to* 1 **do**

    $\mathrm{d}\boldsymbol{G}^{[l-1]} = (\boldsymbol{W}^{[l+1]})^{\mathrm{T}} \cdot (\mathrm{d}\boldsymbol{Z}^{[l+1]})$;

    $\mathrm{d}\boldsymbol{Z}^{[l]} = \mathrm{d}\boldsymbol{G}^{[l]} * \sigma'(\boldsymbol{Z}^{[l]})$;

    $\mathrm{d}\boldsymbol{W}^{[l]} = \frac{1}{N_T}\left[\mathrm{d}\boldsymbol{Z}^{[l]} \cdot (\boldsymbol{G}^{[l-1]})^{\mathrm{T}}\right]$;

    $\mathrm{d}\boldsymbol{b}^{[l]} = \frac{1}{N_T}\sum_{i=1}^{N_T}\mathrm{d}\boldsymbol{Z}^{[l]}$;

  end

  Adam optimization (Kingma & Ba 2014) ;

  for $l = 1$ *to* L **do**

    $\boldsymbol{V}_{dW^{[l]}} = \beta_1\boldsymbol{S}_{dW^{[l]}} + (1-\beta_1)\,\mathrm{d}\boldsymbol{W}^{[l]}$; $\quad \boldsymbol{V}_{db^{[l]}} = \beta_1\mathbf{S}_{db^{[l]}} + (1-\beta_1)\,\mathrm{d}\boldsymbol{b}^{[l]}$;

    $\boldsymbol{S}_{dW^{[l]}} = \beta_2\boldsymbol{S}_{dW^{[l]}} + (1-\beta_2)(\mathrm{d}\boldsymbol{W}^{[l]})^2$; $\quad S_{db^{[l]}} = \beta_2\boldsymbol{S}_{db^{[l]}} + (1-\beta_2)(\mathrm{d}\boldsymbol{b}^{[l]})^2$;

    $\boldsymbol{V}_{dW^{[l]}} := \frac{\boldsymbol{V}_{dW^{[l]}}}{1-\beta_1^{iter}}$; $\quad \boldsymbol{V}_{db^{[l]}} := \frac{V_{db^{[l]}}}{1-\beta_1^{iter}}$; $\quad \boldsymbol{S}_{dW^{[l]}} := \frac{\boldsymbol{S}_{dW^{[l]}}}{1-\beta_2^{iter}}$; $\quad S_{db^{[l]}} := \frac{S_{db^{[l]}}}{1-\beta_2^{iter}}$;

    $\boldsymbol{W}^{[l]} := \boldsymbol{W}^{[l]} - \alpha\frac{\boldsymbol{V}_{dW^{[l]}}}{\sqrt{\boldsymbol{S}_{dW^{[l]}}}+\epsilon}$;

    $\boldsymbol{b}^{[l]} := \boldsymbol{b}^{[l]} - \alpha\frac{V_{db^{[l]}}}{\sqrt{S_{db^{[l]}}}+\epsilon}$;

  end

end

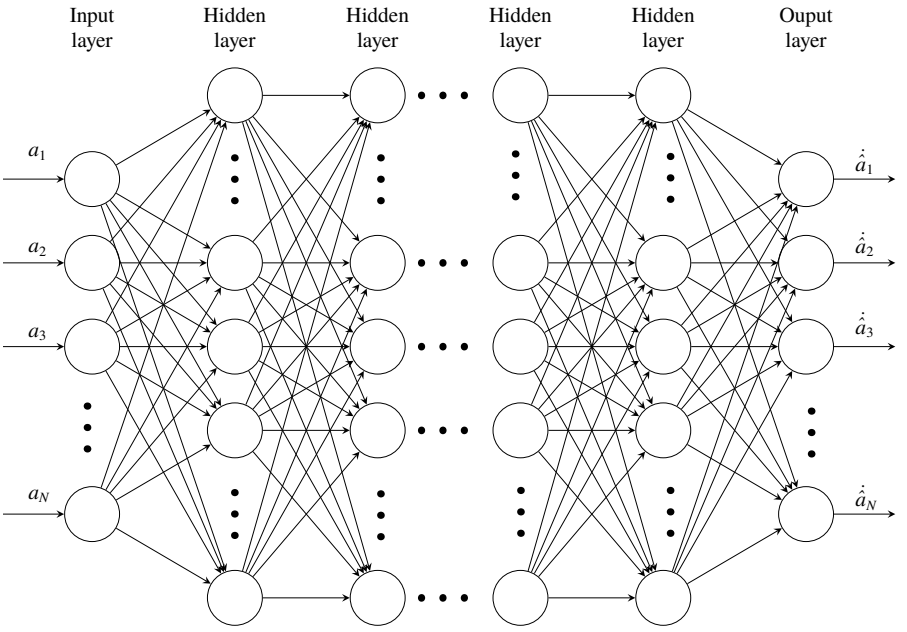

FIGURE 2. An example of a deep feedforward network with N inputs, several hidden layers (HL), and one output layer with N outputs.

system of coupled ODEs (2.12) can now be given by

$$\frac{\mathrm{d}\hat{\boldsymbol{a}}(t)}{\mathrm{d}t} = \boldsymbol{W}^{[L]}\boldsymbol{G}^{[L-1]} + \boldsymbol{b}^{[L]} = \boldsymbol{F}(\boldsymbol{a}(t)), \tag{2.16}$$

where L is the index of the last layer, $\boldsymbol{G}^{[L-1]}$ is the activation function matrix of the penultimate layer, $\boldsymbol{W}^{[L]}$ is the weight matrix of the last layer, and $\boldsymbol{b}^{[L]}$ is the bias vector of the last layer.

This system of coupled ODEs is integrated using an explicit five stage 4th-order Runge–Kutta scheme derived by Kennedy, Carpenter & Lewis (1999). As we can see in algorithm 1, $\boldsymbol{G}^{[L-1]}$ depends on the weights $\boldsymbol{W}$ and biases $\boldsymbol{b}$ from previous layers. Thus for each stage of the 4th-order Runge–Kutta, we need to perform forward propagation to obtain $\boldsymbol{F}(\boldsymbol{a}(t))$. As we have the temporal coefficients $\boldsymbol{a}(t)$, one can reconstruct the flow field using (2.3). However, we are interested in using a ROM in circumstances other than simply reproducing the training data. The approach presented in this work allows us to predict the flow field beyond the training window because $\bar{\boldsymbol{q}}(\boldsymbol{x})$ and $\boldsymbol{\phi}(\boldsymbol{x})$ depend only on the spatial coordinates $\boldsymbol{x}$ and they are calculated using only the training data.

2.3. *Data-driven ROMs*

As previously discussed in § 1, the construction of ROMs is an active area of research and different methodologies for the development of ROMs are available. For example, POD–Galerkin based methods are directly related to the physics of the problem through the projection of the Navier–Stokes equations into a system of ordinary differential equations. Galerkin projection methods require the treatment of the linear

and nonlinear spatial terms appearing in the FOM. Another issue with these methods relates to their expensive application for dynamical systems with strong nonlinearities where one should employ, for example, hyper-reduction techniques (Chaturantabut & Sorensen 2010; Carlberg *et al.* 2011; Zimmermann & Willcox 2016).

In the current approach, we employ DNNs for the regression step in a context similar to the SINDy algorithm. Both the DNN and the original SINDy approaches are data-driven methods and, instead of having a direct connection with the physics of the problem, these techniques learn from data. An advantage of these methods is that the nonlinearities of the problem are considered in the temporal derivatives of the primitive variables, which are obtained from the FOM. Therefore, neither the SINDy nor the DNN method require the treatment of spatial derivatives.

In general, the resulting ROMs constructed via DNNs are not (directly) physically interpretable due to the nonlinearity of the matrix of features and their weights and biases, which are not sparse. We hope that interpretability of complex models can be improved by new topics of research such as 'accountable machine learning' (Navarro *et al.* 2018). The physics of systems with complicated models can also be interpreted by the structure of the state space, e.g. fixed points, Lyapunov exponents, fractal dimensions, bifurcations and DNN models can also be useful for computing such measures. Learning from models simply uses the fact that many realizations of the physical system can be computed very quickly to explore parts of the state space which are not contained in the training data. In some cases, such as the flow past a cylinder, models constructed using sparse regression could directly lead to physically interpretable results (Brunton *et al.* 2016). As expected, the application of DNNs for the regression step adds a penalty cost compared to sparse regression. However it is shown in this work that, for the cases analysed, models obtained using neural networks present better long-term predictive capabilities compared to sparse regression. Both these issues will be discussed in the results section.

3. Hyperparameter optimization

The performance of the DNN described in algorithm 1 depends dramatically on the selection of hyperparameters such as the network depth, L, number of hidden units for each layer, $n^{[l]}$, regularization parameter, λ, and learning rate α. Indeed, finding an optimal set of hyperparameters which minimizes the loss function over a hyperparameter space is a challenging task given the substantial number of free parameters involved.

The manual search, grid search, random search (Bergstra & Bengio 2012) and Bayesian optimization (Brochu, Cora & Freitas 2010) are the most widely used procedures for the hyperparameter optimization. The manual search consists of a direct human trial and error procedure in the search for an optimal configuration of hyperparameters. This procedure is entirely based on prior experience of the user and there is a high probability that an optimal set of hyperparameters is not found. However, a manual search is still useful if the effect of a specific hyperparameter on the model performance can be monitored 'on the fly'. In the grid search method, several combinations of hyperparameter values are tested in a range evenly spaced. This method is extremely time-consuming because the number of trials increases exponentially with the number of hyperparameters. In a random search, one randomly selects each hyperparameter from a defined range and evaluates the model performance. It is time-consuming when a high-dimensional hyperparameter space is analysed, however, Bergstra & Bengio (2012) empirically show that a random

search outperforms a grid search for the hyperparameter optimization both in terms of computational time and model performance.

One of the recent strategies to find an optimal set of hyperparameters is the Bayesian optimization. It is a technique that involves constructing a probabilistic surrogate model to the data in order to determine the most promising hyperparameters to evaluate. Snoek, Larochelle & Adams (2012) showed that Bayesian optimization was able to find optimal hyperparameters for a three-layer convolutional neural network considerably faster than previous approaches and outperformed the state of the art performance at selecting the set of hyperparameters on the CIFAR-10 data set (Krizhevsky 2009).

In this work we use two hyperparameter optimization strategies: random search and Bayesian optimization. For random search, the model generation procedure is presented in the following algorithm form 2. Likewise, Bayesian optimization has the same inputs as random search. To report the performance of each model from a set of candidates, we compute the mean absolute error (MAE) over the training data. The candidate models with lower MAE values are most likely those which will provide the 'best' models. It is possible, however, that the model with the lowest MAE suffers from overfitting.

The current metric does not assess the generalization of the model. However, it is the only one available since we cannot split our data into training and validation sets. The validation data set should provide an unbiased evaluation of the ROM. In our case, we employ the temporal coefficients of the POD modes for the training stage of the model. These modes are computed using a correlation of different snapshots and if we construct the ROM using POD modes obtained with data including the validation set, our model would use a biased set for the training stage. This occurs because the POD correlation matrix would be computed for snapshots of both the training and validation sets. In order to overcome this issue, we reconstruct the flow field of the candidate models with lowest MAEs and then compute $E_r = \|\boldsymbol{q}_{FOM} - \boldsymbol{q}_{ROM}\|$ over the validation set to select the best candidate model. This procedure could even be improved if E_r were computed for all models. However, the computational cost would considerably increase if such metric were employed.

An alternative to the current MAE procedure is to use Akaike's information criterion (AIC) (Akaike 1973) or the Bayesian information criterion (BIC) (Schwarz 1978) as the model selection criteria. These methods try to balance the quality of the fit and model complexity and the main advantage is that there is no need for a validation set. The downside is that AIC and BIC impose a penalty for model complexity which is related to the number of parameters. This can be a problem when dealing with DNNs due to their large number of parameters.

In our experiments, we noticed that a few of the hyperparameters employed in algorithm 1 need to be tuned for the best performance of the DNN. These are listed in algorithm 2. The remaining hyperparameters are determined according to the following procedures in order to reduce the hyperparameter search space. For instance, hyperparameters related to the Adam optimization, such as the exponential decay rates β_1 and β_2, and the constant for numerical stability ϵ, are set as described by Kingma & Ba (2014). The learning rate α and the number of iterations n_{iter} are chosen via manual search. Further details regarding this process can be found in Goodfellow *et al.* (2016, p. 295). As can be observed in the results section, we use the same values of the previous hyperparameters for all flow simulations studied in this work and these values should serve as references for other flows. Following Goodfellow *et al.* (2016), we provide table 1 with information of the hyperparameters

Hyperparameter	Improves performance if	Reason	Warning
Number of hidden units, n_{hidden}	Increased	Increasing the number of hidden units augments the capacity of the model to represent more complex functions.	Increasing this parameter may cause overfitting to the training data.
Number of layers, n_{layers}	Increased	Same as above.	Same as above. One should also be aware that a DNN with a large number of layers and a very small number of hidden units will not work properly.
Regularization parameter, λ	Reduced	Reducing the regularization parameter allows larger weights for the model features. One should expect that some of these features are the most relevant for the model.	Reducing the regularization parameter causes the model to be more prone to overfitting to the training data.
Learning rate, α	Tuned	If α is too small, the optimization process can be slow. If α is too high, the optimization method may lead to overshoot of local minima. This parameter is chosen by monitoring the learning curve.	If α is too large, the learning curve will show strong oscillations. If it is too small, the learning curve may stuck with a high value of the cost function.
Number of iterations, n_{iter}	Tuned	The number of iterations is strictly related to the learning rate. It is chosen by monitoring the learning curve.	As the number of iterations increases, the model goes from underfitting to optimal and, then, to overfitting the training data.

TABLE 1. Effects of hyperparameters on model performance.

presented in algorithm 2 based on their effects on model performance. This table also presents a similar discussion for the hyperparameters chosen via manual search. We expect that it serves as a guideline for choosing the range of values to be explored for the hyperparameters presented in algorithm 2.

Here, we consider that the activation function is also a hyperparameter since it plays a key role in the DNN performance. There are several available activation functions such as sigmoid, hyperbolic tangent (tanh), rectified linear unit (ReLU) (Nair & Hinton 2010), exponential linear unit (ELU) (Clevert, Unterthiner & Hochreiter 2015), to name a few. For general regression problems, tanh and ELU are the most popular functions since they possess nonlinear properties and are continuously differentiable. Clevert *et al.* (2015) show that the ELU function reduces the vanishing gradient effect since its positive part returns the identity. Thus, in the positive part, the derivative is unitary and it is not contractive. On the other hand, tanh is contractive almost everywhere. Furthermore, in our experiments the ELU function provided results with

fewer iterations than a corresponding tanh network and, therefore, we use ELU as the activation function.

Algorithm 2: Model generation procedure

Input : Number of candidate models n_{models}, minimum number of layers $n_{layers_{min}}$, maximum number of layers $n_{layers_{max}}$, number of inputs n_{inputs} , minimum number of hidden units $n_{hidden_{min}}$, maximum number of hidden units $n_{hidden_{max}}$, minimum order of magnitude of the regularization parameter θ_{min}, and maximum order of magnitude of the regularization parameter θ_{max}.

Output : Network depth L, number of hidden units for each layer $n^{[l]}$ and regularization parameter λ

for $i = 1$ *to* n_{models} **do**
  $\lambda_i = 10^{-rand(\theta_{min},\theta_{max})}$; ▷ Regularization parameter
  $L_i = int(rand(n_{layers_{min}}, n_{layers_{max}}))$; ▷ Network depth
  $n_i^{[1]} = n_{inputs}$; $n_i^{[L]} = n_{inputs}$; ▷ Number of units for the input and output layers
  for $l = 2$ *to* $L_i - 1$ **do**
    $n_i^{[l]} = int(rand(n_{hidden_{min}}, n_{hidden_{max}}))$; ▷ Number of hidden units for layer l
  end
end

4. Results

In this section, the present method is first tested in the reconstruction of the dynamics of a damped cubic oscillator. Then, we evaluate the capability of the DNN-ROMs to reproduce the dynamics of a compressible flow past a cylinder including its noise generation. A comparison between the current DNN technique against sparse regression is also presented for the transient regime of an incompressible flow past a cylinder. Finally, the method is employed to create a ROM of a turbulent flow involving the dynamic stall of a plunging airfoil. Again, the DNN solution is compared against that obtained by sparse regression. In each example, we show the ability and limitations of the methods to identify the dynamics of the different nonlinear systems comparing the ROM and FOM solutions.

4.1. *Nonlinear oscillator*

For this first example, we consider a canonical problem in system identification (Brunton *et al.* 2016; Rudy *et al.* 2018): the two-dimensional nonlinear oscillator. The system dynamics are given by

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} -0.1 & 2 \\ -2 & -0.1 \end{bmatrix} \begin{bmatrix} x_1^3 \\ x_2^3 \end{bmatrix}. \tag{4.1}$$

with initial conditions $\begin{bmatrix} x_1 & x_2 \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} 2 & 0 \end{bmatrix}^{\mathrm{T}}$.

We generate 4000 snapshots from $t = 0$ to $t = 40$ by integrating equation (4.1) using an explicit five stage 4th-order Runge–Kutta scheme (Kennedy *et al.* 1999) with a time step of $h = 0.01$. The training window spans the period $0 \leqslant t \leqslant 10$ and the remaining data is used as the test set. The system reconstruction is obtained following

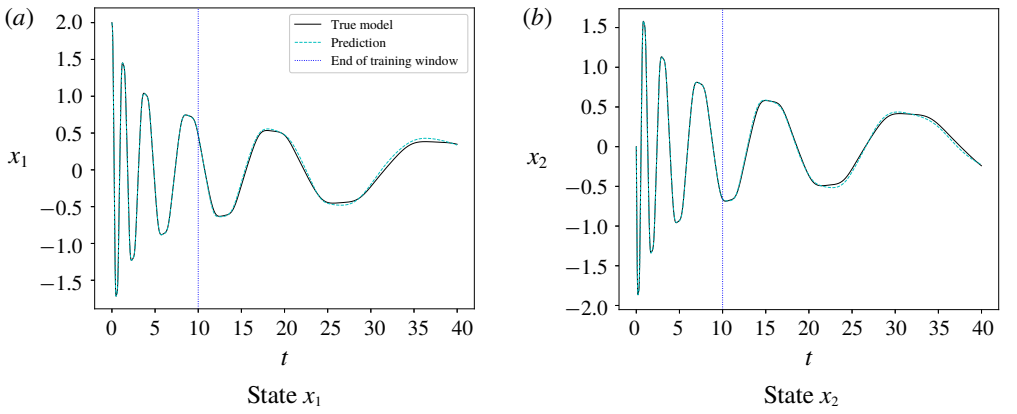

FIGURE 3. (Colour online) Comparison between true model (solid black line) and ROM prediction (dashed cyan line) for nonlinear oscillator.

n_{models}	$n_{layers_{min}}$	$n_{layers_{max}}$	$n_{hidden_{min}}$	$n_{hidden_{max}}$	θ_{min}	θ_{max}
300	3	10	8	36	2	6

TABLE 2. Model generation parameters for ROM of nonlinear oscillator.

DNN architecture	σ	α	λ	n_{iter}	β_1	β_2	ϵ
$2-10-2$	ELU	0.001	1.4062×10^{-6}	20 000	0.9	0.999	1.0×10^{-8}

TABLE 3. Best set of hyperparameters for ROM of nonlinear oscillator.

the procedures defined in § 2. For this first example, it is not necessary to use POD due to the low dimensionality of the system. The parameters employed in algorithm 2 are presented in table 2 and the best set of hyperparameters obtained via random search is listed in table 3. Figure 3 presents the solutions obtained by the FOM (true model) and ROM for the damped cubic oscillator. Results show that the proposed algorithm accurately reproduces the system dynamics during the training window and beyond for the test set.

4.2. *Construction of ROMs for fluid flow simulations*

For the following investigations of the flow past a cylinder and the dynamic stall of a plunging SD7003 airfoil, the system dynamics are modelled by the compressible Navier–Stokes equations in two dimensions and three dimensions, respectively. To simulate the airfoil undergoing a prescribed motion, the equations are solved in a non-inertial frame. In this form, source terms emerge from the grid curvature and frame movement. Here, all terms are solved in the full contravariant form to allow the use of a curvilinear coordinate system $\{\xi^1, \xi^2, \xi^3\}$. For a frame of reference with varying linear velocity, the equations reduce to

$$\frac{\partial}{\partial t}(\sqrt{g}\rho) + \frac{\partial}{\partial \xi^i}(\sqrt{g}\rho u^i) = 0, \tag{4.2}$$

$$\frac{\partial}{\partial t}(\sqrt{g}\rho u^i) + \frac{\partial}{\partial \xi^j}[\sqrt{g}(\rho u^i u^j - \tau^{ij} + g^{ij}p)] + \left\{ \begin{matrix} i \\ jk \end{matrix} \right\} \sqrt{g}(\rho u^k u^j + g^{jk}p - \tau^{kj}) = \sqrt{g}\rho \ddot{h}^i, \tag{4.3}$$

and

$$\frac{\partial}{\partial t}(\sqrt{g}E) + \frac{\partial}{\partial \xi^j}\left\{ \sqrt{g}\left[(E+p)u^j - \tau^{ij}g_{ik}u^k - \frac{\mu}{Re\,Pr}g^{ij}\frac{\partial T}{\partial \xi^i} \right] \right\} = \rho\sqrt{g}(h^j + u^j)g_{ji}\ddot{h}^i. \tag{4.4}$$

The set of equations above represent the continuity, momentum and energy equations. In order to close the system of equations the following relations are employed:

$$E = \frac{p}{\gamma - 1} + \frac{1}{2}\rho u^i g_{ij} u^j + \frac{1}{2}\rho \dot{h}^i g_{ij} \dot{h}^i, \tag{4.5}$$

$$\tau^{ij} = \frac{\mu}{Re}\left(g^{jk}u^i{}_{|k} + g^{ik}u^j{}_{|k} - \frac{2}{3}g^{ij}u^k{}_{|k} \right), \tag{4.6}$$

and

$$h = h_o \sin(kt). \tag{4.7}$$

Here, ρ represents the density, u^i the ith component of the contravariant velocity vector and p is the pressure. The term h is the frame position (cross-stream motion of the plunging airfoil), E is the total energy, μ is the dynamic viscosity and T is the temperature. The dots represent temporal derivatives of the frame position, i.e., frame velocity and acceleration. The aforementioned terms are non-dimensionalised by freestream quantities such as density ρ_∞ and speed of sound c_∞. The length scales are made dimensionless by the cylinder diameter or airfoil chord. In the above equations, g_{ij} and g^{ij} are the covariant and contravariant metric tensors, $\sqrt{g}$ is the Jacobian of the covariant metric tensor, and $\left\{ \begin{matrix} i \\ jk \end{matrix} \right\}$ represents the Christoffel symbols of the second kind. Further details regarding the present formulation can be found in Aris (1989).

The numerical scheme for the spatial discretization of the equations is a high-resolution sixth-order accurate compact scheme (Nagarajan, Lele & Ferziger 2003) implemented on a staggered grid. A sixth-order compact interpolation scheme is also employed (Lele 1992) to obtain fluid properties on the different nodes of the staggered configuration. Due to the non-dissipative characteristics of compact finite-difference schemes, numerical instabilities may arise from mesh non-uniformities, interpolations and boundary conditions and, hence, they have to be filtered to preserve stability of the numerical schemes. The high wavenumber compact filter presented by Lele (1992) is applied in flow regions far away from solid boundaries at prescribed time intervals in order to control numerical instabilities.

The time integration of the fluid equations is carried out by the fully implicit second-order scheme of Beam & Warming (1978) in the near-wall region in order to overcome the time step restriction due to the usual near-wall fine-grid numerical stiffness. A third-order Runge–Kutta scheme is used for time advancement of the equations in flow regions far away from solid boundaries. No-slip adiabatic wall boundary conditions are applied along the solid surfaces and characteristic plus sponge conditions are applied in the far field. For the study of dynamic stall, periodic

n_{models}	$n_{layers_{min}}$	$n_{layers_{max}}$	$n_{hidden_{min}}$	$n_{hidden_{max}}$	θ_{min}	θ_{max}
500	6	10	10	64	2	6

TABLE 4. Model generation parameters for ROM of compressible flow past a cylinder.

DNN architecture	σ	α	λ	n_{iter}	β_1	β_2	ϵ
$10-31-51-25-32-42-51-27-26-10$	ELU	0.001	4.4085×10^{-5}	10 000	0.9	0.999	1.0×10^{-8}

TABLE 5. Best set of hyperparameters for ROM of compressible flow past a cylinder.

boundary conditions are applied along the spanwise direction of the airfoil. In this case, we employ an implicit large eddy simulation (LES) to study the flow physics of deep dynamic stall over a plunging SD7003 airfoil profile, i.e., no subgrid scale model is used. The numerical tool has been previously validated for several simulations of unsteady compressible flows (Wolf 2011; Wolf, Azevedo & Lele 2012*a*; Wolf *et al.* 2012*b*; D'Lelis *et al.* 2019).

4.2.1. *Flow past a cylinder*

In this case, the FOM is obtained by solving the compressible Navier–Stokes equations as detailed in § 4.2. The numerical simulations are conducted for Reynolds and Mach numbers $Re = 150$ and $M = 0.4$, respectively. These dimensionless parameters are computed based on freestream quantities. The grid configuration consists of a body-fitted O-grid with 421×751 points in the streamwise and wall-normal directions, respectively.

The flow is recorded for 1120 snapshots with dimensionless time steps of $h = 0.05$. The snapshots are collected after an initial transient period of the simulation is discarded. The ROM is obtained following the procedure described in § 2 using the pressure norm for the POD correlation matrix. It is important to mention that the use of a kinetic energy norm produced similar results for this case. The training data comprises the first 280 snapshots of the FOM and the remaining data is employed as the test set. The set of hyperparameters employed in algorithm 2 is listed in table 4 and the best set of these parameters obtained via random search is presented in table 5. For this case, Bayesian optimization was also able to produce accurate models but at a higher computational cost compared to random search. One can see that the best architecture of the neural network for this case has 10 layers. However, we also found several other stable and accurate models with six layers, for example. The fact that the best model was found with 10 layers is a coincidence.

Figures 4 and 5 show contours of z-vorticity and divergence of velocity, respectively, along the cylinder and wake regions at time $t = 410$, which is beyond the training window. The snapshots allow a comparison of the results between the FOM and ROM using 2 and 10 POD modes out of 280 modes. Hence, the flow is reconstructed using 0.7 % and 3.5 % of the total information available from the FOM. Although the current simulation is performed for a compressible flow at $M = 0.4$ and $Re = 150$, we could verify that the POD spatial eigenfunctions are almost identical to those from Noack *et al.* (2003), which were obtained for an incompressible flow and $Re = 100$. A video comparing the FOM and ROM solutions is provided in the supplementary material (movie 1) available at https://doi.org/10.1017/jfm.2019.358. Reconstruction of the individual flow variables with 2 POD modes could recover between 50 % and

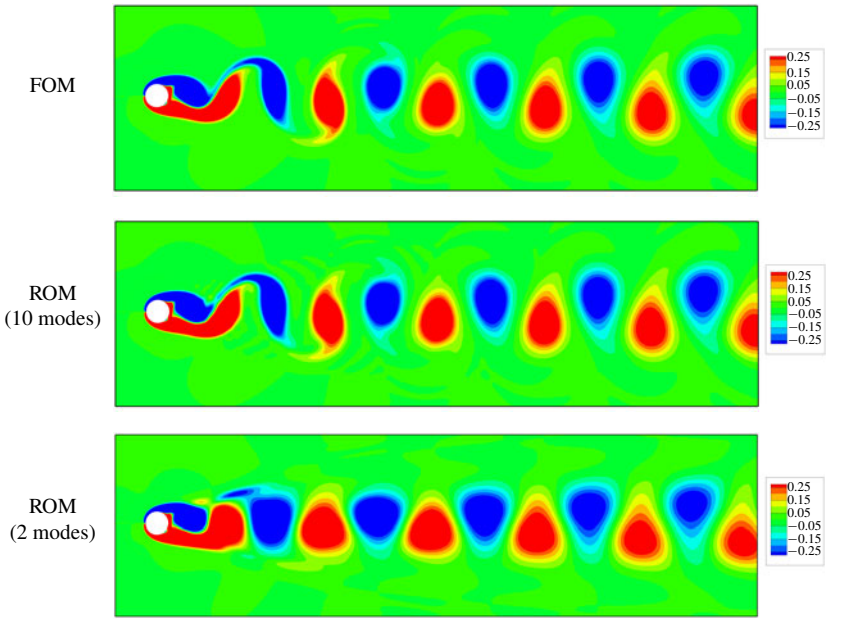

FIGURE 4. (Colour online) Contours of z-vorticity, $t = 410$.

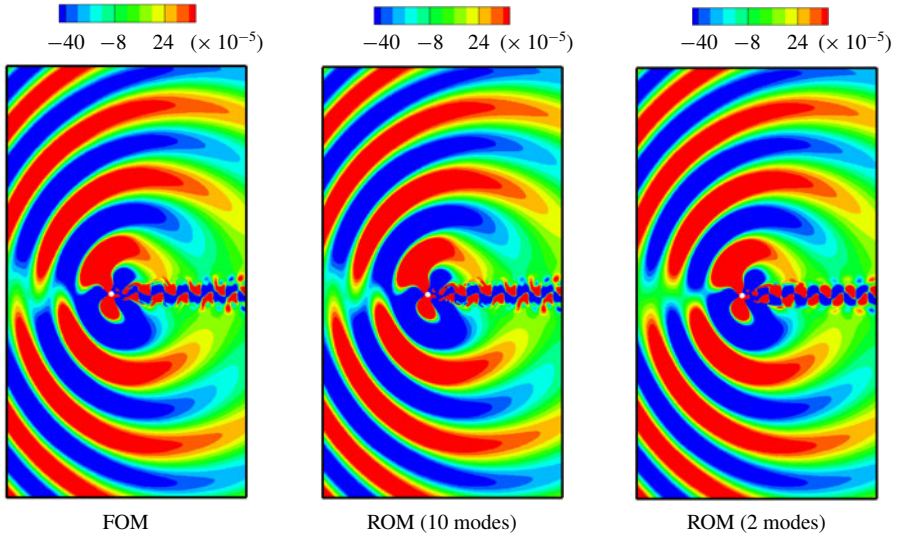

FIGURE 5. (Colour online) Contours of divergence of velocity, $t = 410$.

80 % of the total modal energy, depending on the variable. For example, density is reconstructed using 50 % of the total energy of the system dynamics while y-momentum is reconstructed with 80 % of the total modal energy. Here we use the term 'modal energy' to refer to the ratio between the sum of N POD eigenvalues used in a particular reconstruction over the entire range N_T of eigenvalues available, $\sum_{i=1}^{N} \lambda_i / \sum_{i=1}^{N_T} \lambda_i$. For 10 POD modes the reconstructions could recover 99 % of the

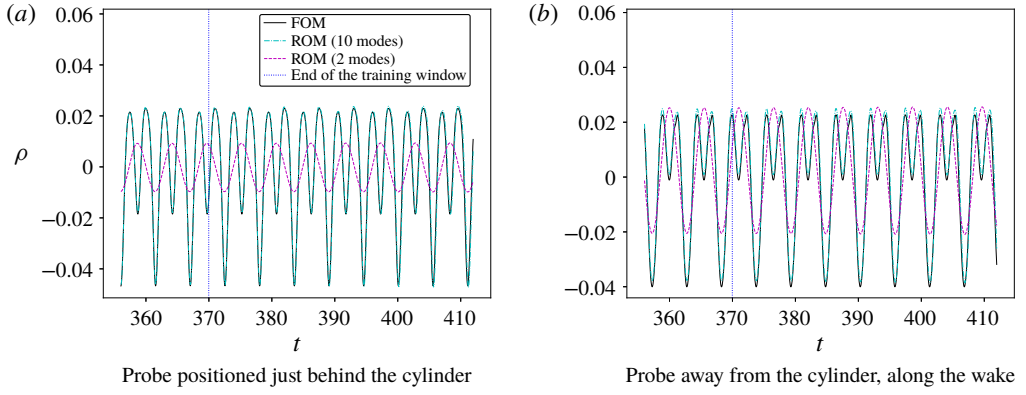

FIGURE 6. (Colour online) Fluctuation time history of density.

energy for all variables and, therefore, should lead to an accurate flow representation. For this particular case, the SPOD technique is not required since the modes are almost monochromatic and, hence, do not require filtering.

One can observe from the figures that the computations of the flow using the ROM framework show good agreement compared to those obtained by the FOM. For the current Reynolds number, the flow develops a typical von Kármán vortex street along the cylinder wake. The periodical pattern of the vortex shedding can be observed in the contours of z-vorticity. Noise generation also occurs in the current unsteady compressible flow simulation. In this case, pressure fluctuations along the cylinder surface are scattered to the far-field and can be observed in the contours of dilatation shown in figure 5. Both the near-field hydrodynamics and the far-field acoustics are recovered by the ROM. The reconstruction using 10 POD modes show excellent agreement with the FOM. When 2 POD modes are employed in the flow reconstruction, discrepancies between the ROM and FOM are evident from the figures. However, the main features of the flow are still recovered by the model. One should note that, despite the use of only two modes, the dynamical system is still stable beyond the training region.

In order to show a more qualitative evaluation of the model reconstructions, the density and x-momentum fluctuation time histories are presented for the FOM and ROMs in figures 6 and 7, respectively. The figures on the left column show results for a probe located just behind the cylinder, close to the surface, at $(x, y) = (0.55, -0.06)$. The cylinder has radius 0.5 and its centre is positioned in the origin of the Cartesian system. On the right column, results are obtained for a probe downstream of the cylinder wake, at $(x, y) = (1.1, -0.06)$. Results are shown for both the training window period and beyond. When two POD modes are employed, the solutions show a less accurate representation of the dynamics observed in the FOM. The density reconstruction is that with the highest discrepancy and that is attributed to the lower energetic content achieved by the first two POD modes. One can notice that the ROM accurately reproduces the FOM results during and beyond the training window when 10 POD modes are employed in the reconstruction.

In order to test the robustness of the method, we employ the DNN approach for the reconstruction of the transient regime of an incompressible cylinder flow. For this study, the two most energetic POD modes containing both the transient and limit cycle dynamics of the flow are obtained from Brunton *et al.* (2016). These modes

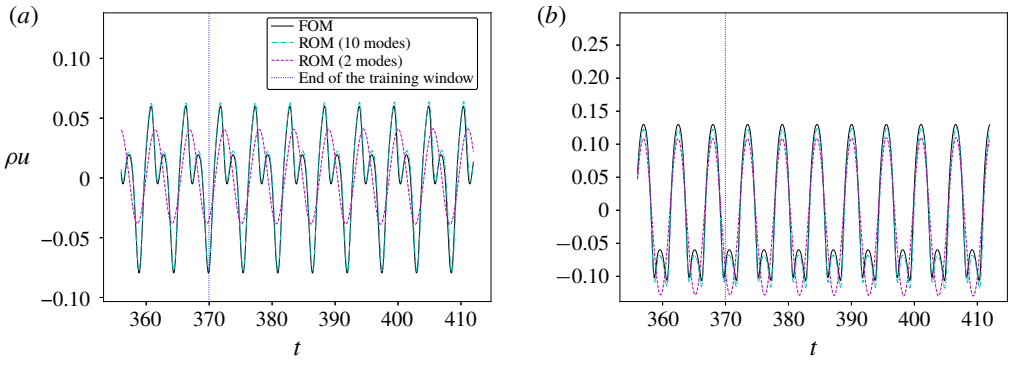

FIGURE 7. (Colour online) Fluctuation time history of x-momentum.

contain both the transient and limit cycle dynamics of the flow. Figure 8 shows the POD temporal modes used for training and testing the DNNs. The first 5000 temporal instants are used to train the models and one can see this data set represented by the black line to the left of the vertical line marking the end of the training window. To the right of the training window, the black line represents the test data which is the correct solution for the temporal dynamics. Figure 8 also shows the results obtained by the current DNN approach and by sparse regression. In the case of sparse regression, we employ the same model obtained by Brunton *et al.* (2016) in table 11 of their supporting information. As one can observe, the DNN is able to recover accurately both the transient and limit cycle solutions of the test data for both POD modes. On the other hand, sparse regression reconstructs the transient portion of the dynamics but not the long-term prediction of the limit cycle. Several models obtained by the DNNs presented similar results compared to those shown in figure 8 and some had similar neural network architectures compared to that of table 5. These results show that the proposed DNN approach has good long-term predictive capabilities and can learn the transient features of the flow.

4.2.2. *Deep dynamic stall of plunging airfoil*

The present study concerns a plunging SD7003 airfoil in deep dynamic stall. The flow conditions have a reduced frequency $k = \pi fL/U_\infty = 0.5$, where f is the plunging frequency, L is the chord of the airfoil and U_∞ is the reference free stream velocity. The motion amplitude is set as $h_o/L = 0.5$ with a static angle of attack $\alpha_0 = 8^\circ$. The chord Reynolds number based on the freestream velocity is $Re = 60\,000$ and the freestream Mach number is $M = 0.1$. This flow condition is relevant for micro air vehicle applications and this case is selected based on the availability of published results from other high fidelity simulations (Visbal 2011) and particle image velocimetry data (Baik *et al.* 2009; Ol *et al.* 2009). In a previous work (D'Lelis *et al.* 2019), we performed a mesh refinement study and validation of the numerical solutions against the references above.

The present mesh configuration consists of a body-fitted O-grid with $441 \times 300 \times 64$ points in the streamwise, wall-normal and spanwise directions, respectively. The grid is generated with 70 % of the surface points located in the suction side of the airfoil to improve the capturing of finer flow scales developing along the turbulent region of

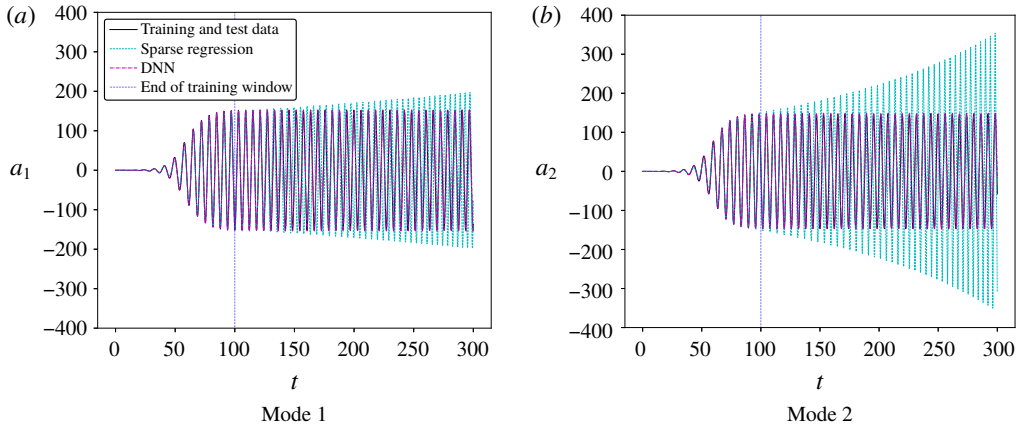

FIGURE 8. (Colour online) Reconstruction of POD temporal modes for transient solution of incompressible flow past a cylinder.

the flow. Due to the favourable pressure gradients in the pressure side of the airfoil, the flow remains laminar along the entire cycle of the plunging motion. The trailing edge of the SD7003 airfoil is rounded in the current simulations with an arc of radius $r/L = 0.0008$. This procedure is required for retaining the smoothness of the metric terms computed by the high-order compact scheme. The spanwise domain is set as $z/c = 0.4$ similarly to Visbal (2011) and the dimensionless time step of the simulation is set as $\Delta t^* = ((\Delta t U_\infty)/L) = 0.00008$.

The plunge motion occurs with an effective angle of attack in the range of $-6^\circ \leqslant \alpha \leqslant 22^\circ$. Defining ψ as the angular position in the plunging cycle, we say that at $\psi = 0^\circ$ the airfoil has no velocity in the y-direction and is at the top-most position of the plunging motion. At $\psi = 90^\circ$ it has the highest velocity in the y-direction downwards and, at $\psi = 180^\circ$, it has no velocity and is at the bottom-most position of the plunging motion. Finally, at $\psi = 270^\circ$ it has the highest velocity in the y-direction upwards. In summary, during the down-stroke instabilities begin to form in the suction side of the airfoil, growing and eventually breaking into finer structures. While this takes place, the dynamic stall vortex forms along the leading edge and is transported through the airfoil suction side increasing the overall lift and creating a nose-down pitching moment. As the leading-edge vortex (LEV) approximates the trailing edge, a trailing-edge vortex (TEV) forms pushing the LEV away from the airfoil. A more complete discussion of the flow dynamics can be found in Visbal (2011) and D'Lelis *et al.* (2019).

Figure 9 shows iso-surfaces of Q-criterion coloured by pressure and it is possible to observe the main flow features described above. In figures 9(*a*) and 9(*b*), it is possible to compare the 3-D solutions of the FOM and ROM, respectively, for the leading edge vortex formation. Figure 9(*c,d*) show a similar comparison for the instant where the trailing edge vortex forms. For both instants, one can observe that the ROM is able to reconstruct the larger scale features of the 3-D flow. A video comparing the FOM and ROM solutions for this flow configuration is provided together with the manuscript (movie 2). The ROM is trained using the first two cycles of the plunging motion and the solutions presented in figure 9 are computed for the fourth cycle, showing that the model is able to reproduce the 3-D flow dynamics beyond the training window. For this study the parameters employed in algorithm 2 are presented in table 6 and

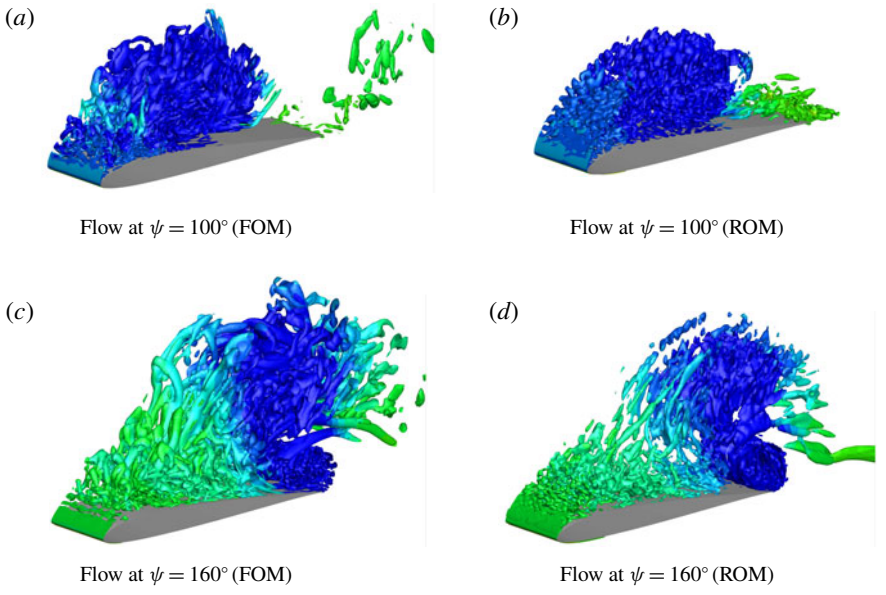

FIGURE 9. (Colour online) Iso-surfaces of Q criterion coloured by pressure at different instants of the plunge motion for the fourth cycle. The ROM was trained using only the first 2 cycles.

n_{models}	$n_{layers_{min}}$	$n_{layers_{max}}$	$n_{hidden_{min}}$	$n_{hidden_{max}}$	θ_{min}	θ_{max}
700	6	12	10	64	2	6

TABLE 6. Model generation parameters for ROM of plunging airfoil in deep dynamic stall.

DNN architecture	σ	α	λ	n_{iter}	β_1	β_2	ϵ
$16-19-52-50-31-16$	ELU	0.001	8.0938×10^{-5}	10 000	0.9	0.999	1.0×10^{-8}

TABLE 7. Best set of hyperparameters for ROM of plunging airfoil in deep dynamic stall (3-D flow).

the best set of hyperparameters, obtained via random search, is listed in table 7. It is important to mention that, for this case, Bayesian optimization was unable to produce stable and accurate models.

The first 16 SPOD modes are employed to reduce the dimensionality of the input data in the 3-D flow reconstruction. Other POD reconstructions were tested with a different number of modes and it was observed that the first 16 modes contained the main features of the leading and trailing edge vortex formation. For example, adding 10 more modes did not improve significantly the solution and, beyond mode 26, the SPOD temporal modes presented complex behaviour, being composed of several frequencies and confusing the training stage of the DNNs. This issue could be improved by running the simulation for a longer period with a lower time step to improve convergence of the POD modes. One should note that the SPOD allows an energy shift of the modes to obtain frequency filtered temporal dynamics. In this

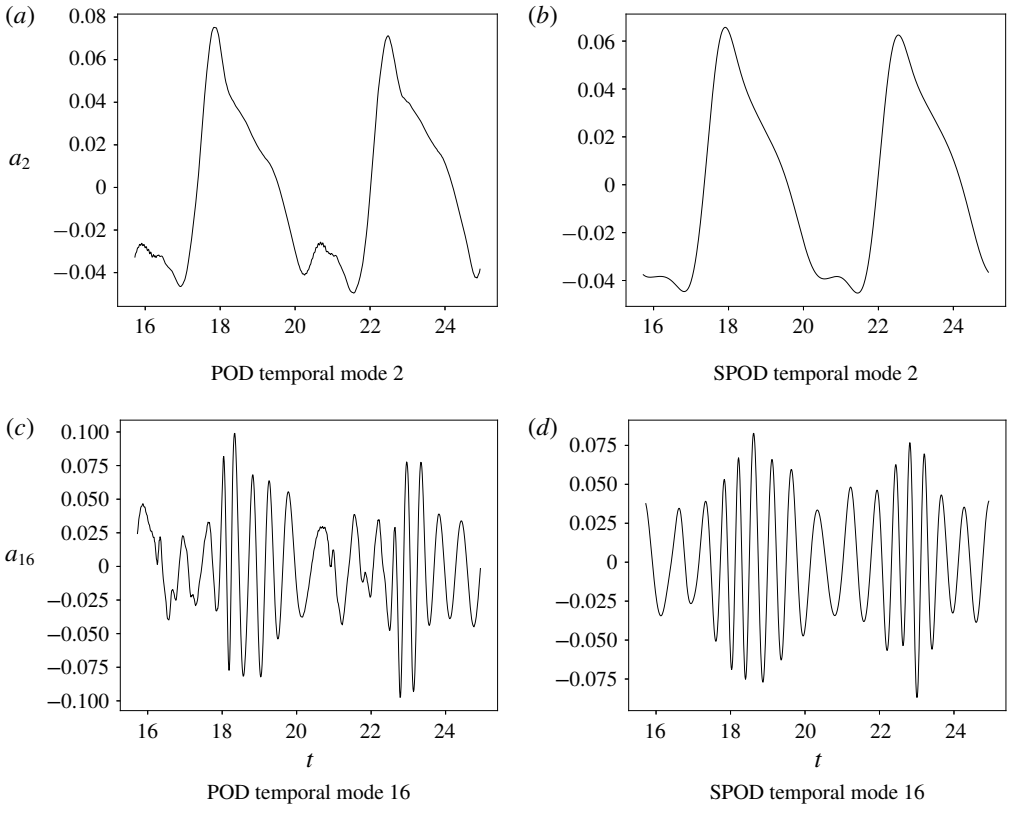

FIGURE 10. Characteristics of POD and SPOD temporal modes for 3-D flow past airfoil under plunging motion.

sense, the high-frequency noise observed in the POD temporal modes is not discarded but is shifted to higher SPOD modes. This procedure allows a better pairing of POD modes if the coherent structures have periodicity. In the present study, due to the unsteady boundary conditions and the lack of symmetry in the flow, we do not expect pairing of the SPOD modes.

For the present turbulent flow, the temporal modes of the standard snapshot POD are composed by several frequencies and, moreover, contain some noise that degrades the training of the DNNs. The SPOD provides the most energetic modes for specific frequency bands, allowing a better identification of the individual modes and smoothing out the temporal coefficients. Here, a box filter is applied to 25 % of the POD correlation matrix. We observed that filter values in the range of 20 % to 40 % provide good results for the model reconstructions. If a lower filter window is employed, noise can still be present in the temporal modes. On the other hand higher filter windows may considerably modify the temporal modes transforming them into quasi-sinusoidal signals. This is undesirable since the relevant dynamics of the flow can be lost and too many SPOD modes may be required for the construction of the ROM. An example of the impact of SPOD on the characteristics of temporal modes 2 and 16 can be seen in figure 10. In this example, the second POD mode exhibits high-frequency noise which is filtered by the SPOD. Meanwhile, POD mode 16 shows a complicated pattern due to the contribution of several frequencies. This

DNN architecture	σ	α	λ	n_{iter}	β_1	β_2	ϵ
$10-46-23-17-19-10$	ELU	0.001	2.3365×10^{-5}	10 000	0.9	0.999	1.0×10^{-8}

TABLE 8. Best set of hyperparameters for ROM of plunging airfoil in deep dynamic stall (spanwise-averaged flow).

temporal coefficient is considerably simplified by the application of SPOD as can be observed in the figure. It is clear that, different from the cylinder case, for the plunging airfoil the POD modes contain more complex dynamics which are composed of multiple Fourier modes.

In order to further evaluate the present DNN approach, the ROM reconstructions are also performed for spanwise-averaged solutions of the flow and the ROM is constructed following the procedures defined in § 2. We employ six plunging cycles (2244 snapshots) for computing the models, discarding initial flow transients. The training data contains the first two cycles (748 snapshots) of the FOM and the remaining data is used as the test set. The parameters employed in algorithm 2 are the same as those presented in table 6 and the best set of hyperparameters, obtained via random search, is listed in table 8. Here, we apply the box filter to 25 % of the POD correlation matrix and employ the first 10 SPOD modes for the ROM construction, which was sufficient to recover the most important features of the dynamic stall solution for the spanwise-averaged study.

Figures 11 and 12 present snapshots of density and x-component of momentum, respectively, for the FOM and ROM. A video comparing these solutions for the spanwise-averaged flow is also provided together with the manuscript (movie 3). The flow features of the dynamic stall can be observed for different stages of the plunging cycle, given by different values of ψ in the figures. It is possible to track the LEV over the suction side of the airfoil. The current ROM is able to recover the important dynamics of the leading edge vortex formation, its transport and ejection, besides the trailing edge vortex formation and ejection. One should note that the results presented in these figures are obtained for a plunging cycle beyond the training region. Therefore, the current ROM is capable of reproducing the flow dynamics for the test set. The high wavenumber features shown in the FOM are not present in the SPOD modes and, hence, they are not recovered by the ROM. If additional SPOD modes were employed in the model reconstruction, these features would appear. However, the overall cost of the simulations would increase considerably if accurate higher POD modes were required. For the present study, we observed that the fine scale flow dynamics are related to higher modes since most of the energetic content is related to the airfoil plunging motion represented by the first POD modes.

Fluctuation time histories are presented in figure 13 for density and x-momentum. These properties are computed by the FOM and ROM at probe locations in the proximity of the leading and trailing edges, at $(x, y) = (0.07, 0.07)$ and $(x, y) = (0.97, 0.07)$, respectively. It is important to mention that the airfoil leading edge is at the origin of the Cartesian system. The plots show the training and test regions demonstrating that the ROM is stable beyond the training set and can accurately reproduce the main dynamics of the flow. The fast oscillations resolved by the FOM are not represented due to the SPOD basis truncation. However, all the relevant features of the flow fluctuations are captured by the DNN model obtained with the first 10 SPOD modes.

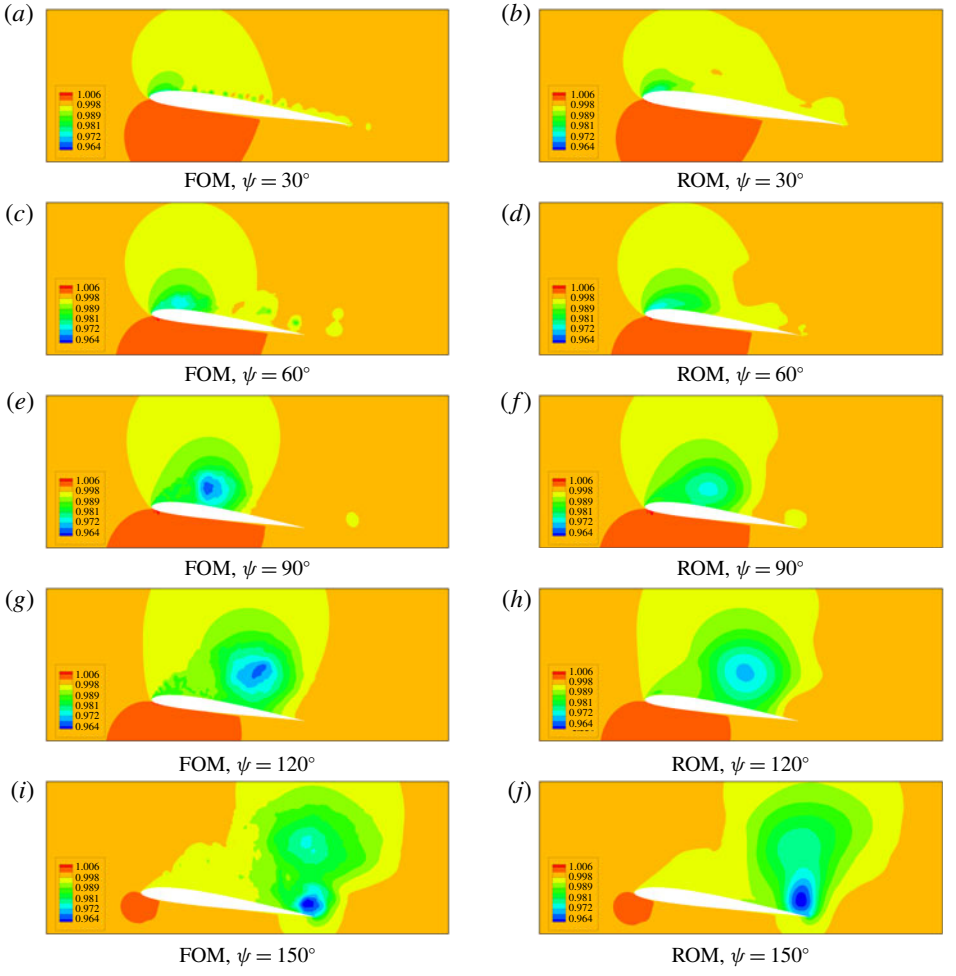

FIGURE 11. (Colour online) Density contours at several phase angles for the fifth plunging cycle.

In figure 14, one can observe a comparison between ROMs built using the current DNN approach and sparse regression. For this case, both the LASSO and Ridge regression techniques were tested with the original SINDy algorithm. Although the LASSO presented a higher computational cost, it provided better models compared to the Ridge regression. For this case, the computational cost of the DNN regression is, on average, 40 times higher than that of SINDy. However, as shown for SPOD modes 1 and 8, the SINDy algorithm was not able to provide stable models. The same can be said for the other SPOD modes. Sparse regression can learn the dynamics during the training window and also for a few cycles in the test set but its solution is not stable for long-time predictions. On the other hand, despite the higher cost, the DNN approach presents good long-time predictive capabilities with stable and accurate solutions beyond the training window.

In order to evaluate the robustness of the present framework, we test the capability of the ROM in reconstructing the flow field using a time step different than that of the simulation. Figure 15 shows the reconstruction of SPOD temporal modes using

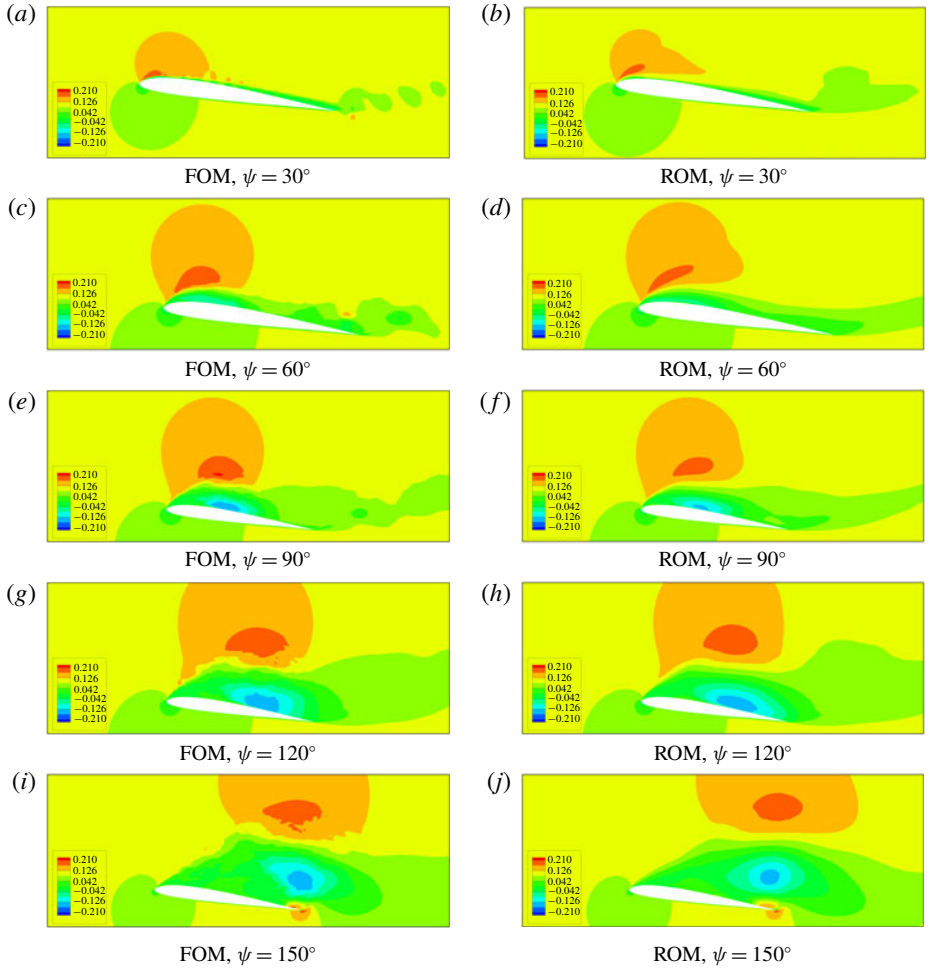

FIGURE 12. (Colour online) Contours of x-component of momentum at several phase angles for the fifth plunging cycle.

different finite difference schemes and time steps. The derivatives of the temporal modes are computed via an explicit second-order scheme and by sixth- and tenth-order compact schemes. When the ROM is constructed using the same time step of the numerical simulation, the first 10 SPOD modes are highly resolved in time. For this case, all three schemes for computing the temporal derivatives provide accurate results and the random search is able to find a set of hyperparameters which results in an accurate ROM. However, when the DNN is trained using SPOD modes obtained by a time step five times larger than that of the simulation, the resolution of the higher POD temporal modes can become compromised. Then, it is important to use the compact schemes to obtain accurate representations of the derivatives. In this case, the random search could only find suitable ROMs using the compact schemes. As shown in figure 15, the models are still stable and reproduce the same dynamics observed with lower time steps. This shows that the current ROM framework is robust and can be constructed using a subset of the data of the FOM, with lower temporal resolution.

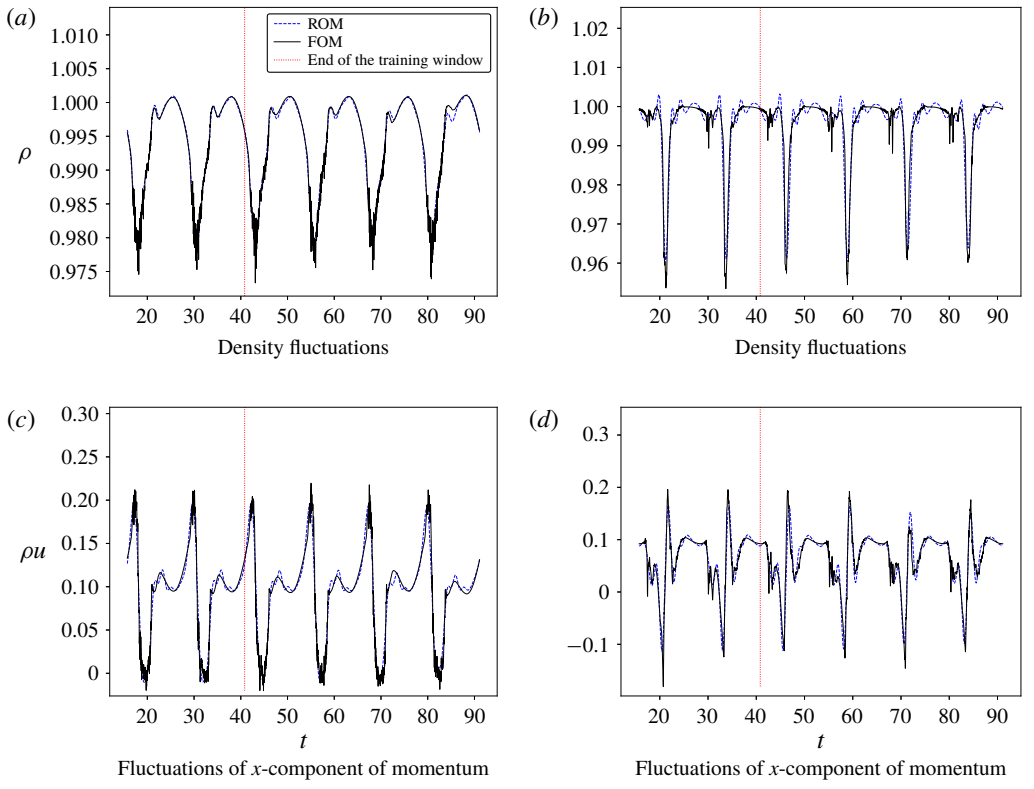

FIGURE 13. (Colour online) Fluctuation time histories computed by the FOM and ROM for probe locations in the proximity of the leading edge (*a*,*c*) and trailing edge (*b*,*d*).

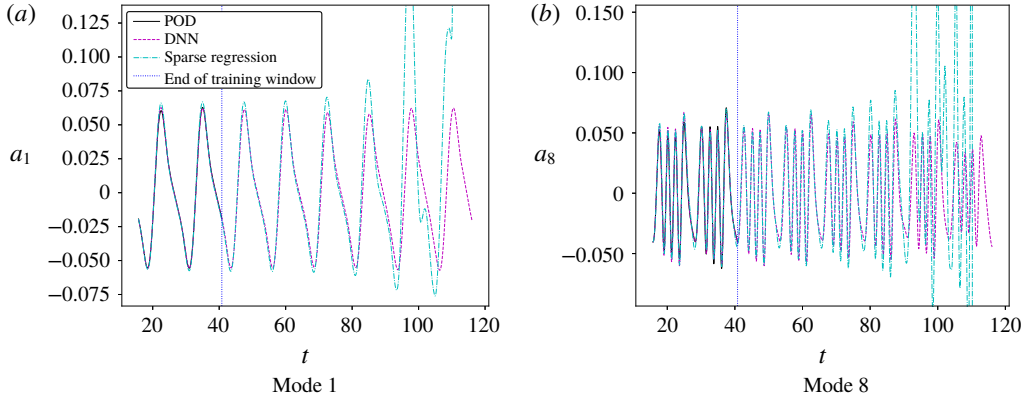

FIGURE 14. (Colour online) Reconstruction of SPOD temporal modes using the current DNN approach and the original SINDy algorithm with LASSO.

Finally, we compute the phase-averaged aerodynamic coefficients for the plunging SD7003 airfoil. Lift and drag coefficients are shown in figure 16 for the FOM and ROM using the six cycles computed in the training and test sets. Both the pressure and friction forces are accounted for in the calculation of the aerodynamic coefficients.

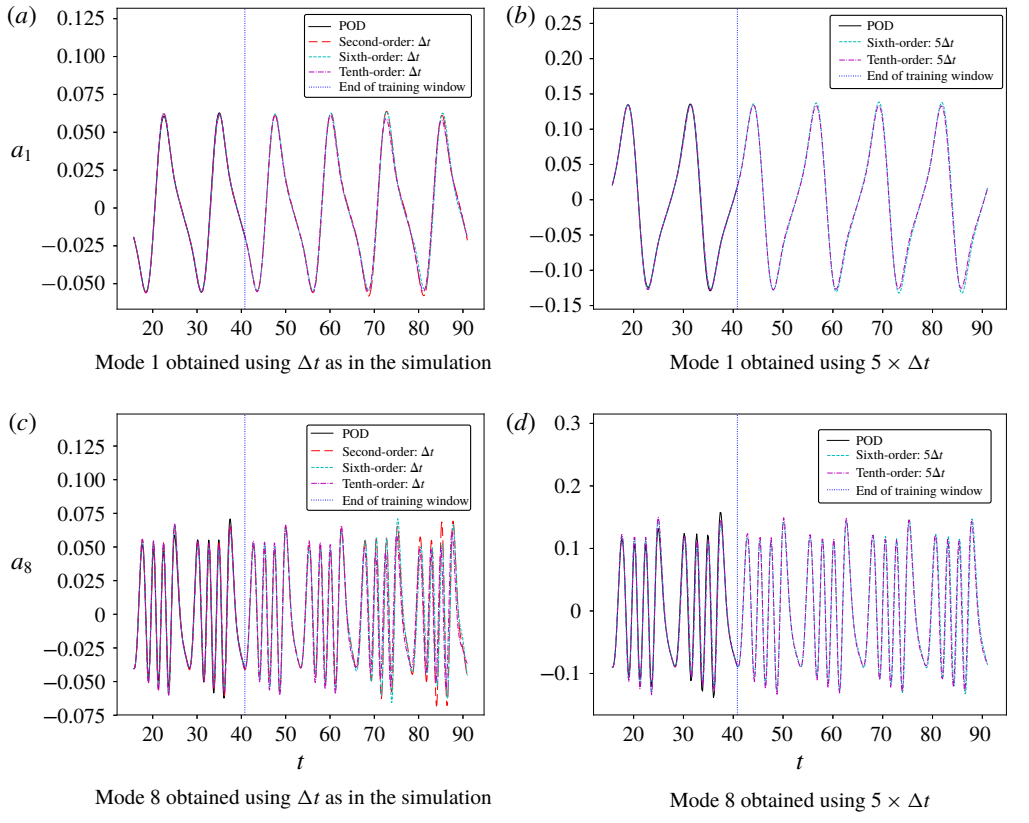

FIGURE 15. (Colour online) Reconstruction of SPOD temporal modes using different finite difference schemes and time steps.

However, for this case the pressure component dominates the aerodynamic loads. The effective incidence angles are depicted in the figures and the aerodynamic coefficients are calculated using the 10th-order compact scheme for the derivatives of the temporal modes and the first 10 SPOD modes. As a validation of the current solutions, the results from Visbal (2011) are also shown. Results are presented for the ROMs computed using both the pressure and kinetic energy norms. The DNNs could find stable and accurate models for both POD norms and the aerodynamic coefficients obtained for the best models are compared in the figure. As one can see, the present ROM solutions show good comparisons to the current LES and Visbal (2011). The total simulation cost of the FOM for this case was 100 000 h. On the other hand, the full cost of the ROM, including the optimization of the hyperparameters, plus the training and evaluation of 700 models, required approximately seven hours. Hence, the computational cost for the training procedure is around 100 models per hour in a single GPU. Once a ROM is chosen the cost of the simulation is reduced to a few seconds.

5. Conclusions

We present a methodology for constructing ROMs combining flow modal decomposition and regression analysis via DNNs. The framework is implemented in a

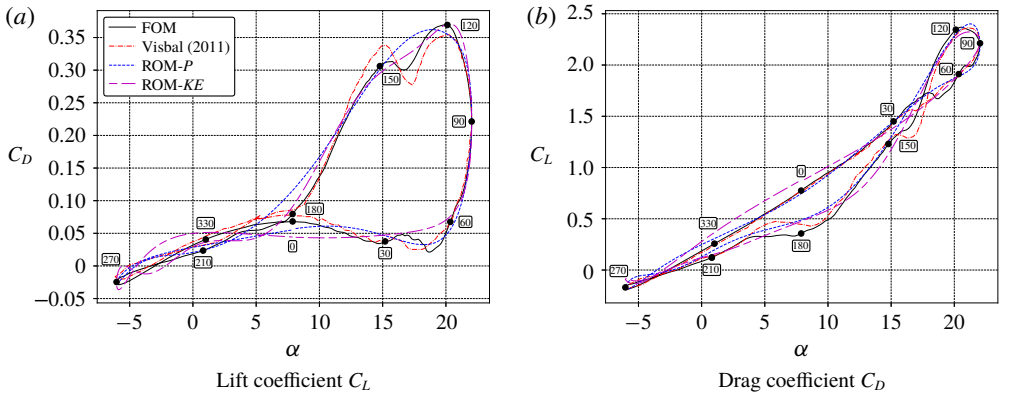

FIGURE 16. (Colour online) Phase-averaged aerodynamic coefficients computed by Visbal (2011), the current LES (FOM), and the DNN approach with POD norms based on pressure (P) and kinetic energy (KE).

context similar to that of the SINDy algorithm recently proposed in the literature. The details of the methodology are described including algorithm charts which facilitate the understanding and implementation of the proposed framework. The source code can be downloaded from http://cces.unicamp.br/software/.

The method is tested for different problems involving nonlinear dynamical systems: the canonical nonlinear damped oscillator, the flow past a circular cylinder at a low Reynolds number and the turbulent flow past a plunging SD7003 airfoil under deep dynamic stall. For the previous two cases, the compressible Navier–Stokes equations are solved in full contravariant form and additional non-inertial terms are added to the equations to simulate the airfoil plunging motion. Numerical simulations are performed using a high-order compact finite difference flow solver. Then, the high fidelity results from the simulations are used as the input data for the construction of the ROMs.

To create stable and accurate ROMs of more complex flows, the application of the SPOD was found necessary. Hence, in order to filter high frequency content present in the temporal modes of the classical snapshot POD, we apply a filter function to the correlation matrix (SPOD approach). The proposed numerical framework allows the prediction of the flow field beyond the training window using larger time increments than those employed by the FOM, which demonstrates the robustness of the current ROMs constructed via DNNs. The resolution of the numerical schemes used for computation of the POD temporal mode derivatives is shown to have an important role when larger time increments are employed in the construction of the ROMs. In this case, the higher POD modes are composed by a broad range of frequencies and the accurate representation of the temporal derivatives is crucial for obtaining ROMs via regression analysis.

A discussion regarding the optimization of hyperparameters for obtaining the best ROMs via DNNs is provided together with a description of the effects of individual hyperparameters on model performance. Here, we test the random search and the Bayesian optimization which are the most widely used procedures for hyperparameter optimization. To report the performance of each model from a set of parameters, we compute the MAE over the training data and choose the candidate models with lower MAE values. Then, the best model is chosen based on the differences between

ROM and FOM solutions over the validation set. Following this approach, the random search produced the best models at lower computational costs for all cases investigated in this work. It is worth mentioning that, in order to reduce the set of parameters for optimization, we first selected the hyperparameters related to the optimization step, such as the learning rate and the number of iterations, by manual search. We also chose the exponential linear unit activation function over the hyperbolic tangent function for the current problems since it provided results with fewer iterations.

Using 10 POD modes, 99 % of the flow energy is recovered in the cylinder flow study. The ROM obtained for this case shows an excellent agreement with the FOM. Even when two POD modes are employed in the reconstruction, the ROM is still stable beyond the training set and captures most of the dynamics of the vortex shedding and sound wave propagation. Both the DNN and the original SINDy approaches are tested for the reconstruction of the transient regime of an incompressible flow past a cylinder. In this case, regression is performed for the two most energetic POD modes of the flow, obtained from Brunton *et al.* (2016). The DNN model is able to accurately recover both the transient and limit cycle solutions of the test data for both POD modes. On the other hand, sparse regression reconstructs the transient portion of the dynamics but not the long-term prediction of the limit cycle.

In the dynamic stall configuration, the flow is turbulent along the airfoil suction side, mostly during the downstroke motion. The complex flow dynamics of this case exhibit unpaired POD modes with high frequency noise. We show how the SPOD approach modifies the temporal modes for this study. Reduced-order models are constructed both for the full 3-D and the spanwise-averaged flow solutions. In both cases, the ROMs are able to capture the dynamics of the leading edge stall vortex, including its formation, transport and ejection, and of the trailing edge vortex. The total simulation cost of the FOM for this case was 100 000 h. On the other hand, the full cost of the ROM was approximately seven hours in a single GPU. This cost already includes the optimization of hyperparameters, plus the training and evaluation of 700 models. When the best model is chosen, the cost of the simulation is reduced to a few seconds. The data obtained by the DNN-ROMs were used to compute aerodynamic coefficients of the dynamic stall and good agreement was found compared to the LES used as the FOM. Again, we compare models obtained by DNNs and sparse regression and show that, despite the higher computational cost, the DNN approach presents good long-time predictive capabilities with stable and accurate solutions beyond the training window. On the other hand, the best model obtained from sparse regression could learn the dynamics during the training window and also for a few cycles in the test set but its solution was not stable for long-time predictions. We expect that, in a future work, the current methodology can be further improved for applications in flow control and extrapolation of flow configurations other than those used for training.

Acknowledgements

The authors acknowledge the financial support received from Fundação de Amparo à Pesquisa do Estado de São Paulo, FAPESP, under grant nos. 2013/08293-7 and 2013/07375-0, and from Conselho Nacional de Desenvolvimento Científico e Tecnológico, CNPq, under grant nos. 304335/2018-5 and 407842/2018-7. The support provided by CNPq through a scholarship for the first author is likewise gratefully appreciated. We also thank CEPID-CeMEAI for providing the computational resources used in this work. We gratefully acknowledge B. D'Lelis and T. Ricciardi for providing the high fidelity dynamic stall and cylinder data employed in the current work.

Supplementary movies

Supplementary movies are available at https://doi.org/10.1017/jfm.2019.358.

REFERENCES

ABADI, M. *et al.* 2016 TensorFlow: Large-scale machine learning on heterogeneous distributed systems. arXiv:1603.04467.

AKAIKE, H. 1973 *Information Theory and an Extension of the Maximum Likelihood Principle*, pp. 199–213. Springer.

ARIS, R. 1989 *Vectors, Tensors, and the Basic Equations of Fluid Mechanics*. Dover.

BAIK, Y. S., RAUSCH, J., BERNAL, L. & OL, M. V. 2009 Experimental investigation of pitching and plunging airfoils at Reynolds number between 1×10^4 and 6×10^4. In *AIAA Paper* 2009-4030.

BALAJEWICZ, M., TEZAUR, I. & DOWELL, E. 2016 Minimal subspace rotation on the Stiefel manifold for stabilization and enhancement of projection-based reduced order models for the compressible Navier–Stokes equations. *J. Comput. Phys.* **321**, 224–241.

BEAM, R. M. & WARMING, R. F. 1978 An implicit factored scheme for the compressible Navier–Stokes equations. *AIAA J.* **16** (4), 393–402.

BENGIO, Y. 2009 Learning deep architectures for AI. *Found. Trends Mach. Learn.* **2** (1), 1–127.

BERGSTRA, J. & BENGIO, Y. 2012 Random search for hyper-parameter optimization. *J. Mach. Learn. Res.* **13**, 281–305.

BROCHU, E., CORA, V. M. & FREITAS, N. 2010 A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. arXiv:1012.2599.

BRUNTON, S. L. & NOACK, B. R. 2015 Closed-loop turbulence control: progress and challenges. *Appl. Mech. Rev.* **67**, 050801.

BRUNTON, S. L., PROCTOR, J. L. & KUTZ, N. J. 2016 Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proc. Natl Acad. Sci. USA* **113** (15), 3932–3937.

CARLBERG, K., BOU-MOSLEH, C. & FARHAT, C. 2011 Efficient non-linear model reduction via a least-squares Petrov–Galerkin projection and compressive tensor approximations. *Intl J. Numer. Meth. Engng* **86** (2), 155–181.

CAZEMIER, W., VERSTAPPEN, R. W. C. P. & VELDMAN, A. E. P. 1998 Proper orthogonal decomposition and low-dimensional models for driven cavity flows. *Phys. Fluids* **10** (7), 1685–1699.

CHATURANTABUT, S. & SORENSEN, D. 2010 Nonlinear model reduction via discrete empirical interpolation. *SIAM J. Sci. Comput.* **32**, 2737–2764.

CLAINCHE, S. & VEGA, J. M. 2017 Higher order dynamic mode decomposition to identify and extrapolate flow patterns. *Phys. Fluids* **29**, 084102.

CLEVERT, D., UNTERTHINER, T. & HOCHREITER, S. 2015 Fast and accurate deep network learning by exponential linear units (ELUs). arXiv:1511.07289.

D'LELIS, B. O. R., WOLF, W. R., YEH, C. & TAIRA, K. 2019 High-fidelity simulation and flow control of a plunging airfoil under deep dynamic stall. *AIAA Paper* 2019-1395.

GOODFELLOW, I., BENGIO, Y. & COURVILLE, A. 2016 *Deep Learning*. MIT Press.

GREEN, M. A., ROWLEY, C. W. & HALLER, G. 2007 Detecting vortex formation and shedding in cylinder wakes using Lagrangian coherent structures. *J. Fluid Mech.* **572**, 111–120.

HALLER, G. 2015 Lagrangian coherent structures. *Annu. Rev. Fluid Mech.* **47**, 137–162.

JUANG, J. N. & PAPPA, R. S. 1985 An eigensystem realization algorithm for modal parameter identification and model reduction. *J. Guid. Control Dyn.* **8**, 620–627.

KENNEDY, C. A., CARPENTER, M. H. & LEWIS, R. M. 1999 Low-storage, explicit Runge–Kutta schemes for the compressible Navier–Stokes equations. *ICASE Report No.* 99-22.

KINGMA, D. P. & BA, J. 2014 Adam: a method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*. arXiv:1412.6980.

KRIZHEVSKY, A. 2009 Learning multiple layers of features from tiny images. Tech. Rep.

KUTZ, J. 2017 Deep learning in fluid dynamics. *J. Fluid Mech.* **814**, 1–4.

LELE, S. K. 1992 Compact finite difference schemes with spectral-like resolution. *J. Comput. Phys.* **103** (1), 16–42.

LING, J., KURZAWSKI, A. & TEMPLETON, J. 2016 Reynolds averaged turbulence modelling using deep neural networks with embedded invariance. *J. Fluid Mech.* **807**, 155–166.

LING, J. & TEMPLETON, J. 2015 Evaluation of machine learning algorithms for prediction of regions of high Reynolds averaged Navier Stokes uncertainty. *Phys. Fluids* **27**, 085103.

LUHAR, M., SHARMA, A. S. & MCKEON, B. J. 2014 On the structure and origin of pressure fluctuations in wall turbulence: predictions based on the resolvent analysis. *J. Fluid Mech.* **751**, 38–70.

LUMLEY, J. L. 1967 The structure of inhomogeneous turbulence. In *Atmospheric Turbulence and Wave Propagation*, pp. 166–178. Nauka.

NAGARAJAN, S., LELE, S. K. & FERZIGER, J. H. 2003 A robust high-order compact method for large eddy simulation. *J. Comput. Phys.* **191** (2), 392–419.

NAIR, V. & HINTON, G. E. 2010 Rectified linear units improve restricted Boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, pp. 807–814. Omnipress.

NAVARRO, L. C., NAVARRO, A. K. W., ROCHA, A. & DAHAB, R. 2018 Connecting the dots: Toward accountable machine-learning printer attribution methods. *J. Visual Commun. Image Represent.* **53**, 257–272.

NOACK, B. R., AFANASIEV, K., MORZYNSKI, M., TADMOR, G. & THIELE, F. 2003 A hierarchy of low-dimensional models for the transient and post-transient cylinder wake. *J. Fluid Mech.* **497**, 335–363.

OL, M. V., BERNAL, L., KANG, C. & SHYY, W. 2009 Shallow and deep dynamic stall for flapping low Reynolds number airfoils. *Exp. Fluids* **46** (5), 883–901.

OLSON, B. J. 2012 Large-eddy simulation of multi-material mixing and over-expanded nozzle flow. PhD thesis, Stanford University.

ÖSTH, J., NOACK, B. R., KRAJNOVI, S., BARROS, D. & BORÉE, J. 2014 On the need for a nonlinear subscale turbulence term in POD models as exemplified for a high-Reynolds-number flow over an Ahmed body. *J. Fluid Mech.* **747**, 518–544.

PAN, S. & DURAISAMY, K. 2018 Long-time predictive modeling of nonlinear dynamical systems using neural networks. arXiv:1805.12547.

PROTAS, B., NOACK, B. R. & ÖSTH, J. 2015 Optimal nonlinear eddy viscosity in Galerkin models of turbulent flows. *J. Fluid Mech.* **766**, 337–367.

RIBEIRO, J. H. M. & WOLF, W. R. 2017 Identification of coherent structures in the flow past a NACA0012 airfoil via proper orthogonal decomposition. *Phys. Fluids* **29** (8), 085104.

ROCKWOOD, M. P., TAIRA, K. & GREEN, M. A. 2016 Detecting vortex formation and shedding in cylinder wakes using Lagrangian coherent structures. *AIAA J.* **55**, 15–23.

ROWLEY, C. W., COLONIUS, T. & MURRAY, R. M. 2004 Model reduction for compressible flows using POD and Galerkin projection. *Physica D: Nonlinear Phenomena* **189** (12), 115–129.

RUDY, S. H., BRUNTON, S. L., PROCTOR, J. L. & KUTZ, N. J. 2017 Data-driven discovery of partial differential equations. *Sci. Adv.* **3** (4), e1602614.

RUDY, S. H., KUTZ, J. N. & BRUNTON, S. L. 2018 Deep learning of dynamics and signal-noise decomposition with time-stepping constraints. arXiv:1808.02578.

SAN, O. & MAULIK, R. 2018 Extreme learning machine for reduced order modeling of turbulent geophysical flows. *Phys. Rev.* E **97**, 042322.

SCHMID, P. J. 2010 Dynamic mode decomposition of numerical and experimental data. *J. Fluid Mech.* **656**, 5–28.

SCHWARZ, G. 1978 Estimating the dimension of a model. *Ann. Stat.* **6**, 461–464.

SHARMA, A. S., MOARREF, R., MCKEON, B. J., PARK, J. S., GRAHAM, M. D. & WILLIS, A. P. 2016 Low-dimensional representations of exact coherent states of the Navier–Stokes equations from the resolvent model of wall turbulence. *Phys. Rev.* E **93**, 021102.

SIEBER, M., PASCHEREIT, C. O. & OBERLEITHNER, K. 2016 Spectral proper orthogonal decomposition. *J. Fluid Mech.* **792**, 798–828.

SIROVICH, L. 1986 Turbulence and the dynamics of coherent structures. Part I. Coherent structures. *Q. Appl. Maths* **45**, 561–571.

SNOEK, J., LAROCHELLE, H. & ADAMS, R. P. 2012 Practical Bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems 25*, pp. 2951–2959. Curran Associates.

STRÖFER, C. M., WU, J. L., XIAO, H. & PATERSON, E. 2019 Data-driven, physics based feature extraction from fluid flow fields using convolutional neural networks. *Commun. Comput. Phys.* **25**, 625–650.

TAIRA, K., BRUNTON, S. L., DAWSON, S. T. M., ROWLEY, C. W., COLONIUS, T., MCKEON, B. J., SCHMIDT, O. T., GORDEYEV, S., THEOFILIS, V. & UKEILEY, L. S. 2017 Modal analysis of fluid flows: an overview. *AIAA J.* **55** (12), 4013–4041.

TOWNE, A., SCHMIDT, O. T. & COLONIUS, T. 2018 Spectral proper orthogonal decomposition and its relationship to dynamic mode decomposition and resolvent analysis. *J. Fluid Mech.* **847**, 821–867.

TU, J. H., ROWLEY, C. W., LUCHTENBURG, D. M., BRUNTON, S. L. & KUTZ, N. J. 2014 On dynamic mode decomposition: theory and applications. *J. Comput. Dyn.* **1**, 391–421.

VISBAL, M. R. 2011 Numerical investigation of deep dynamic stall of a plunging airfoil. *AIAA J.* **49**, 2152–2170.

VLACHAS, P. R., BYEON, W., WAN, Z. Y., SAPSIS, T. P. & KOUMOUTSAKOS, P. 2018 Data-driven forecasting of high-dimensional chaotic systems with long short-term memory networks. *Proc. R. Soc. Lond.* A **474** (2213).

WAN, Z. Y., VLACHAS, P., KOUMOUTSAKOS, P. & SAPSIS, T. 2018 Data-assisted reduced-order modeling of extreme events in complex dynamical systems. *PLOS ONE* **13** (5), 1–22.

WANG, J. X., WU, J. L. & XIAO, H. 2017 Physics-informed machine learning approach for reconstructing Reynolds stress modeling discrepancies based on DNS data. *Phys. Rev. Fluids* **2**, 034603.

WOLF, W. R. 2011 Airfoil aeroacoustics: LES and acoustic analogy predictions. PhD thesis, Stanford University.

WOLF, W. R., AZEVEDO, J. L. F. & LELE, S. K. 2012*a* Convective effects and the role of quadrupole sources for aerofoil aeroacoustics. *J. Fluid Mech.* **708**, 502–538.

WOLF, W. R., LELE, S. K., JOTHIPRASARD, G. & CHEUNG, L. 2012 Investigation of noise generated by a DU96 airfoil. *AIAA Paper* 2012-2055.

XAVIER, G. & YOSHUA, B. 2010 Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics* (ed. Y. W. Teh & M. Titterington).

ZIMMERMANN, R. & WILLCOX, K. 2016 An accelerated greedy missing point estimation procedure. *J. Sci. Comput.* **38**, 2827–2850.