Aerospace Science and Technology 47 (2015) 406–419

Contents lists available at ScienceDirect

Aerospace Science and Technology

www.elsevier.com/locate/aescte

Nonlinear aerodynamic reduced order modeling by discrete time recurrent neural networks

Andrea Mannarino *, Paolo Mantegazza

Dept. of Aerospace Science and Technology, Politecnico di Milano, Via La Masa 34, 20156 Milano, Italy

ARTICLE INFO

Article history:
Received 3 May 2013
Received in revised form 13 October 2015
Accepted 14 October 2015
Available online 11 November 2015

Keywords:
Recurrent neural networks
Limit cycle oscillation
Transonic aerodynamics
Nonlinear aeroelasticity

ABSTRACT

Nowadays, viable estimations of transonic aerodynamic loads can be obtained through the tools of computational fluid dynamics. Nonetheless, even with the increasing available computer power, the cost of solving the related non-linear, large order models still impedes their widespread use in conceptual/preliminary aircraft design phases, whereas the related nonlinearities might critically affect design decisions. Therefore, it is of utmost importance to develop methods capable of providing adequately precise reduced order models, compressing large order aerodynamic systems within a highly reduced number of states. This work tackles such a problem through a discrete time recursive neural network formulation, identifying compact models through a training based on input–output data obtained from high-fidelity simulations of the aerodynamic problem alone. The soundness of such an approach is verified by first evaluating the aerodynamic loads resulting from the harmonic motion of an airfoil in transonic regime and then checking aeroelastic limit cycle oscillations inferred from such a reduced neural system against high fidelity response analyses.

1. Introduction

Nowadays, a viable estimation of transonic aerodynamic loads acting on flying airplanes is often provided by computational fluid dynamics (CFD) codes, so allowing to adequately tackle aircraft stability and response analyses, for both flight mechanics [1,2] and aeroelastic [3,4] applications. However, such simulations are still computationally expensive, being characterized by a large number of unknowns and often limited to the most significant validation cases [3,5,6].

In order to introduce the typical nonlinear effects encountered in transonic flows, researchers have focused some of their efforts toward the development of reduced order models (ROMs). These compact system representations are designed to maintain an accuracy as close as possible to that of their parent high-fidelity aerodynamic analyses. An extensive overview of these methods can be found in [6] and references therein. The techniques mainly employed in the literature for the generation of reduced order models can be roughly subdivided in three main branches.

The first is the group of subspace projection techniques, such as the proper orthogonal decomposition [7,8], and, in a generalized sense, the harmonic balance method [9]. These approaches project the high fidelity model into a subspace spanned by a very efficient basis, which is able to represent any solution of interest through a small number of states. With proper orthogonal decomposition-based techniques, the related numerical bases are computed mostly through the singular value decomposition of a snapshot matrix, whose columns are time samples of very accurate responses to well chosen forcing terms [10]. The harmonic balance method on the other hand considers directly a truncated Fourier series as reduced order basis, limiting its application to periodic solutions [11,12].

The second branch is related to the adoption of generalized interpolation methods, e.g. radial basis function or Kriging interpolators [13,14]. Such methods employ a high-fidelity system for pointwise evaluations of its response, while a high order interpolation is applied for computing the response at any intermediate points of interest. Therefore, this ROM works as a general nonlinear input–output mapping, permitting to represent the dynamic system analytically. Even if it is a robust technique, its application seems limited to the evaluation of nonlinear aeroelastic systems stability, as demonstrated in the cited references.

The third group is represented by identification techniques based on input–output data pairs. The Volterra series method [6], i.e. the generalization of the impulse response to nonlinear systems, belongs to this group. Another approach, the one followed in this work, is characterized by the adoption of neural networks.

* Corresponding author.
E-mail address: andrea.mannarino@polimi.it (A. Mannarino).

http://dx.doi.org/10.1016/j.ast.2015.10.009
1270-9638/

Nomenclature

b	airfoil/wing semi-chord
c	airfoil chord
C_L, C_M	coefficients of lift and moment
$\mathbf{e}$	network output error
h, θ	plunge and pitch degree of freedoms
$k = \frac{\omega c}{V_\infty}$	reduced frequency
m	airfoil/wing mass
$r_\theta^2 = \frac{J_\theta}{mb^2}$	nondimensional airfoil/wing moment of inertia
$\mathbf{u}$	network input
$V^* = \frac{V_\infty}{\omega_\theta b\sqrt{\mu}}$	reduced velocity
$\mathbf{W}^a$, $\mathbf{W}^b$, $\mathbf{W}^c$	network synaptic weights
$x_\theta = \frac{S_\theta}{mb}$	nondimensional airfoil/wing static unbalance
$\mathbf{x}$	network state
$\mathbf{y}$	network output
$\mathbf{\Lambda}$	network Jacobian matrix
$\mu = \frac{m}{\pi\rho_\infty b^2}$	fluid-mass ratio
ρ_∞	fluid density
$\tau_s = \omega_\theta t$	structural adimensional time
$s = \frac{V_\infty t}{b}$	aerodynamic adimensional time
$\Phi(v)$	network activation function
ω_h, ω_θ	uncoupled plunging and pitching circular frequencies
CFD	Computational fluid dynamics
DTRNN	Discrete time recurrent neural network
LCO	Limit cycle oscillation
ROM	Reduced order model

Recently, discrete time recurrent neural networks have been employed in the order reduction of relatively simple aeroelastic systems [15–17]. In particular, the first two references employ a neural system with radial basis functions as computational units, within a framework that can be interpreted as a system identification based on a nonlinear autoregression with exogeneous input [18]. Reference [17] instead employs a support vector machine in the identification of unsteady aerodynamic loads. This technique has shown promising results in various machine learning applications and seems to have found its way also in problems where compact system representations are required.

Particular emphasis will be given in the present work to the determination of limit cycle oscillation (LCO) solutions of nonlinear aeroelastic systems.

In the context of the related theory, an LCO is a dynamic bifurcation. The reader can find a vast supporting literature on the analysis of all the different bifurcation types, regarding generic nonlinear systems [19,20] and aeroelastic applications [5,21]. A few details pertaining to the aeroelastic case are considered here.

Within the framework of fluid–structure interaction, LCOs may be driven by aerodynamic nonlinearities, and the related behavior can be associated to the formation of large vortical flow structures, as in the case of low speed, high angle of attack flow regimes [22,23], or to complex shock motions in transonic flows, even when using Euler flow models [5,24,25]. In this last case, which is of main interest in this work, a nonlinear aerodynamic model would allow the simulation of this phenomenon.

Such a moving shock wave may undergo very large displacements, eventually disappearing and reappearing during an LCO period [5,9]. Because of the fact that a shock wave introduces a discontinuity in the flow field, this kind of behavior can be assumed as dynamically nonlinear [24].

Also structural nonlinearities can lead to LCOs, whether the flow is transonic or not, as presented in [23,26,27], but the study of this kind of phenomena is not of interest here.

Aeroelastic limit cycles are usually determined in numerical experiments by time marching integrations [5,6]. Such methods seem to be used mainly to validate the stability changes predicted by Hopf bifurcation analyses with varying dynamic pressure, leading to stable/unstable responses or LCOs. Here instead, the system bifurcation point will be identified through free responses calculations, checking a posteriori if the system is asymptotically stable around the origin or if its behavior converges toward an LCO.

In this work a discrete time recurrent neural network (DTRNN) in state-space form [28] is used to identify nonlinear aerodynamic responses and compute aeroelastic limit cycle oscillations. Such formulation permits to consider the state and the input of the network only one step behind the current state, without keeping the old values of the input (and output in the case of references [15–17]) of several previous time steps in memory.

The present effort has multiple goals: present a novel, neural network-based ROM technique in the discrete time domain, analyze the performance of this methodology in Euler-based aerodynamic loads identification, perform nonlinear aeroelastic simulations comparing the results with the related high fidelity outcomes and determine the ROM sensitivity to parameter changes.

The work is organized as follows. In Section 2.1 the CFD solver employed is presented and all its main features are detailed. In Section 2.2 an introduction to neural networks terminology and to its recurrent framework for dynamic systems modeling is provided. Section 2.3 details the training algorithm used to optimize the network parameters in order to predict any response of interest. Section 3 presents in detail the results obtained for two standard test cases: an airfoil oscillating in pitch and a two degree-of-freedom typical section undergoing limit cycle oscillations due to large shock wave motion. Finally, in Section 4 the most interesting findings of this work are resumed.

2. Numerical methodology

2.1. Aerodynamic solver

For a high fidelity modeling of the aerodynamic problem, the in-house solver AeroFoam developed at Politecnico di Milano [29] is chosen. This application is supported by OpenFOAM libraries for the management of the mesh data, the computation of the numerical solution and the pre/post-processing phases. It is a Reynolds-Averaged Navier Stokes (RANS) density-based solver for aero-servo-elastic applications, written exploiting the Arbitrary-Lagrangian-Eulerian formulation for moving grids. It is a finite volume, cell-centered solver, that can treat both structured and unstructured grids. In the present computations, the Euler flow model is chosen, therefore the effects of viscosity and thermal conductivity will be neglected.

AeroFoam is the first density-based RANS solver implemented within the framework of OpenFOAM, realized to overcome the limits of built-in pressure-based solvers in the transonic regime, e.g. sonicFoam, because their non-conservative formulation does not permit to solve accurately transonic and supersonic regimes.

Regarding the present inviscid application, the convective fluxes are discretized by the classical Roe's approximated Riemann solver, which is a first order, monotone scheme, blended by the centered approximation provided by the Lax–Wendroff scheme, resulting in a second order, high-resolution scheme. The spatial discretization is completed by the entropy fix of Harten and Hyman and the flux limiter by van Leer [30].

The time discretization is performed by an explicit five-stage Runge–Kutta scheme, which presents a first order convergence. Dual time stepping and a full approximation storage multi-grid technique are combined to speed up the convergence of time-accurate simulations.

An extended illustration of the aeroelastic capabilities of AeroFoam can be found in [3,29].

As will be shown in the next sections, this CFD solver will be used to generate the input–output time histories required for tuning the neural model employed in this work. In the present applications, the input will be represented by the structural motion, while the outputs will be the associated aerodynamic loads.

2.2. Recurrent neural networks

A neural network is a massively parallel distributed process made up of simple processing units, the neurons. Using an analogy with our brain, a large number of interconnected neurons would have the natural capability of learning new rules through experience, using them when needed. This knowledge is acquired through a learning process, and stored in the synaptic connections linking the neurons. Neural networks have been widely used in nonlinear system identifications because of their abilities in self-learning, adaptivity and nonlinear modeling.

As demonstrated in the literature [31], neural models are a powerful tool for approximating nonlinear dynamic systems, even when the system itself is unknown and only the input–output data are available. Therefore, they permit a sort of black-box modeling of any nonlinear system, avoiding the burden of formulating a structured parametrization of the equations describing the physical model. In fact, when using neural networks, the model structure is determined only by the layout of the network connections, and the related parameters are determined either through experimental or computational models, thus requiring none or a very little prior knowledge at most.

A neural network is generally composed by an input layer, an arbitrary number of hidden layers and an output layer.

The input layer receives the input data from the external environment and passes it to the computational kernel represented by the neurons. These units receive a linear combination of the input, whose coefficients are called synaptic weights. This signal is passed through the neurons, which are modeled by a nonlinear squashing map, e.g. a logistic or hyperbolic tangent function. If the network is used to approximate a nonlinear process, then its computational layer should be hidden, meaning that the output of its neurons should not be the direct output of the network. Thus the real output is usually a linear combination of the outcome of each neuron, presented to the external environment by the output layer.

For modeling nonlinear dynamic systems, a memory effect should be introduced, therefore a recurrent scheme must be adopted. In this way the output of each neuron is fed back as a new input, after the application of a time delay. A logical scheme of such a network is shown in Fig. 1.

The related mathematical model is given by:

$$\begin{cases} \mathbf{x}_{n+1} &= \boldsymbol{\Phi}\left(\mathbf{W}^a \mathbf{x}_n + \mathbf{W}^b \mathbf{u}_n\right) \\ \mathbf{y}_n &= \mathbf{W}^c \mathbf{x}_n \end{cases} \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^{n_x}$ is the network state, $\mathbf{u} \in \mathbb{R}^m$ is the external input, $\mathbf{y} \in \mathbb{R}^p$ is the output, $\boldsymbol{\Phi} : \mathbb{R}^{n_x} \longrightarrow \mathbb{R}^{n_x}$ is the set of activating functions, and $\mathbf{W}^a \in \mathbb{R}^{n_x \times n_x}$, $\mathbf{W}^b \in \mathbb{R}^{n_x \times m}$ and $\mathbf{W}^c \in \mathbb{R}^{p \times n_x}$ are the matrices containing the synaptic weights, with n_x, m and p being the state, input and output space dimensions respectively. As can be seen by the framework proposed in Fig. 1, only one hidden layer will be used in the present work. This architecture has demonstrated to possess good convergence properties while being described by a limited number of neurons [18,31].

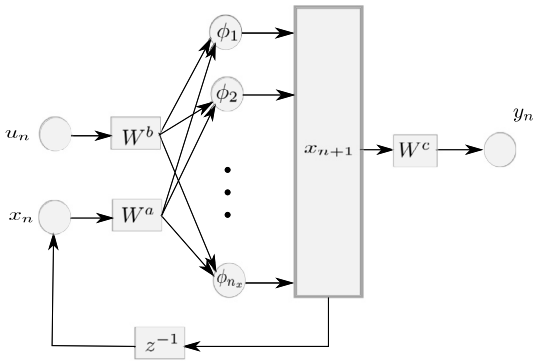


Fig. 1. Example of DTRNN.

From Eq. (1) can be noted that the identified model is assumed to be represented by a strictly proper system, i.e. the output $\mathbf{y}_n$ does not depend on the input $\mathbf{u}_n$ explicitly. The related output matrix is structured as:

$$\mathbf{W}^c = \begin{bmatrix} \mathbf{I}_{p \times p} & \mathbf{0}_{p \times (n_x - p)} \end{bmatrix} \tag{2}$$

where $\mathbf{I}$ is the identity matrix, $\mathbf{0}$ the null matrix, whose dimensions are shown by their subscripts.

The structure of Eq. (2) points out that the outputs of the first p neurons are also those of the network. This framework allows to assign physically meaningful initial conditions to Eq. (1), as will be seen in the next section.

Following a few preliminary numerical simulations, used to test the ability of different sigmoid functions to represent the system nonlinearities of interest, the hyperbolic tangent has been found to be the best activation function in the considered applications. Its mathematical expression is:

$$\phi(v) = \tanh(v) \qquad \phi'(v) = \operatorname{sech}^2(v) \tag{3}$$

where v is the neuron potential. Along with its definition, also its first derivative is provided, because it plays a significant role within the training algorithm as will be seen in the next section.

2.3. Training procedure

The most demanding effort in setting up a meaningful neural network model is, without a doubt, played by its training. Different strategies have been proposed in the literature, in particular for DTRNNs, such as the back propagation through time (BTTP) algorithm, for off-line learning, and the real time recurrent learning (RTRL) algorithm, for on-line implementations. Both algorithms have demonstrated good convergence properties within the realm of the applications of interest of the present work [28,32].

Nonetheless, especially for systems characterized by strong nonlinearities, it has been verified that a simple gradient descent-based learning, e.g. the approach followed by the BTTP and RTRL algorithms previously cited, is somewhat inefficient [33]. Thus more robust optimization algorithms should be used, such as the Levenberg–Marquardt (LM) optimization method [34] or a genetic algorithm (GA) [35].

In fact, the training of a neural network can be viewed as a nonlinear least squares optimization problem, which can greatly benefit from the knowledge of the system Jacobian matrix.

Such a matrix is referred to the derivative of the system output with respect to the parameters to be designed for minimizing a given cost function, i.e. in this case the synaptic weights of Eq. (1). To such an aim let us define the state Jacobian matrix:

$$\mathbf{\Lambda}_n = \frac{\partial \mathbf{x}_n}{\partial \boldsymbol{\theta}} \tag{4}$$

where $\boldsymbol{\theta} = \left(\text{vec}\left(\mathbf{W}^a\right)^T, \text{vec}\left(\mathbf{W}^b\right)^T\right)^T$, $\text{vec}(\cdot)$ being the operator ordering a matrix into a vector by stacking its columns. Because it is associated to a dynamic system, the above Jacobian matrix will be time dependent. As will be shown, this quantity can be computed through a simple finite difference approximation of Eq. (4) or by the analytical development carried out in this section.

The network training aims at finding out the optimal synaptic weight values that minimize a given figure of merit, that in our case is a quadratic function of the output error $\mathbf{e}(t)$, defined as:

$$\mathbf{e}_n = \hat{\mathbf{y}}_n - \mathbf{W}^c \mathbf{x}_n, \quad \text{with} \quad \mathbf{e}_n = \mathbf{e}(t_n) \tag{5}$$

where $\hat{\mathbf{y}}(t)$ is the output of the reference high fidelity model. The associated cost function reads as:

$$F = \frac{1}{2}\sum_{n=1}^{N_t} \mathbf{e}_n^T \mathbf{e}_n \tag{6}$$

where N_t is the number of sampling points. The optimization algorithm will find at least a local minimum of this function with respect to the unknown optimal synaptic weights.

It should be noted that, despite its relatively robust convergence properties, the LM solver may not converge when started from an initial guess point $\boldsymbol{\theta}_0$ too far away from the optimal solution.

For this reason, a hybrid technique is implemented here, running a few iterations of global GA first, resuming LM when the genetic solution hooks up an optimum region, that in this work is characterized by a cost function value smaller than a selected threshold, fixed to 10, chosen after several runs. In the present applications, such a value assures a good and fast convergence of the LM algorithm in the refinement stage. In fact, beside fostering convergence from very rough initial guesses, an added advantage of such a hybrid approach resides in avoiding the calculation of almost useless initial Jacobian matrices, resuming their calculation when LM provides its full advantage of a faster convergence.

In relation to the LM algorithm used in the refinement stage, it becomes useful to define the vector:

$$\mathbf{E}(\boldsymbol{\theta}) = \left(\mathbf{e}_1(\boldsymbol{\theta})^T \quad \mathbf{e}_2(\boldsymbol{\theta})^T \cdots \mathbf{e}_{N_t}(\boldsymbol{\theta})^T\right)^T \tag{7}$$

with $\mathbf{E} \in \mathbb{R}^{N_t \cdot p}$.

The associated Jacobian matrix, defined as:

$$\mathbf{J}(\boldsymbol{\theta}) = \frac{\partial \mathbf{E}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \tag{8}$$

with dimensions $\mathbb{R}^{(N_t \cdot p) \times n_\theta}$, and split into the blocks of dimension $\mathbb{R}^{p \times n_\theta}$:

$$\mathbf{J}(\boldsymbol{\theta}) = \left[\mathbf{J}_1(\boldsymbol{\theta})^T \quad \mathbf{J}_2(\boldsymbol{\theta})^T \cdots \mathbf{J}_{N_t}(\boldsymbol{\theta})^T\right]^T \tag{9}$$

can be computed through a simple finite difference discretization of Eq. (8):

$$\mathbf{J}_n = \frac{\mathbf{e}_n(\boldsymbol{\theta}_k) - \mathbf{e}_n\left(\boldsymbol{\theta}_{k-1}\right)}{\theta_{i,k} - \theta_{i,k-1}}, \qquad n = 1, \ldots, N_t \tag{10}$$

where $\theta_{i,k}$ is the value of the i-th element of the parameter vector describing the network at the k-th iteration of the LM method.

On the other hand, if the knowledge of the network dynamics, represented by Eq. (1), wants to be exploited, then each block is defined as:

$$\mathbf{J}_n(\boldsymbol{\theta}) = \frac{\partial \mathbf{e}_n(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = -\mathbf{W}^c \mathbf{\Lambda}_n(\boldsymbol{\theta}) \qquad n = 1, \ldots, N_t \tag{11}$$

Having defined the state Jacobian matrix in Eq. (4), it is easy to see that its dimensions are given by a matrix belonging to $\mathbb{R}^{n_x \times n_\theta}$, with $n_\theta = n_x \cdot (n_x + m)$ according to the dimensions defined in Eq. (1). For a more straightforward computation of its elements, it is further split in the blocks:

$$\mathbf{\Lambda}_n = \left[\frac{\partial \mathbf{x}_n}{\partial W_{ij}^a} \quad \frac{\partial \mathbf{x}_n}{\partial W_{ij}^b}\right] \tag{12}$$

Remembering the size of each vector/matrix involved and the order they have been sorted with, their assembly will result in a straightforward operation. Defining the single-entry matrix $\mathbf{I}_{ij}$, as the matrix with a 1 in the position (i, j) and zero elsewhere and using the shorthand notation $\mathbf{z}_n = \mathbf{W}^a \mathbf{x}_n + \mathbf{W}^b \mathbf{u}_n$, the direct computation of the blocks of Eq. (12) from Eq. (1) reads:

$$\frac{\partial \mathbf{x}_{n+1}}{\partial W_{ij}^a} = \mathbf{\Phi}'(\mathbf{z}_n)\left(\mathbf{I}_{ij}\mathbf{x}_n + \mathbf{W}^a \frac{\partial \mathbf{x}_n}{\partial W_{ij}^a}\right) \qquad \begin{matrix} i = 1, \ldots, n_x \\ j = 1, \ldots, n_x \end{matrix} \tag{13}$$

with $\mathbf{\Phi}'(\mathbf{z}_n) = \text{Diag}_{n_x}\left[\phi_1'(z_{1,n}), \phi_2'(z_{2,n}), \cdots, \phi_{n_x}'(z_{n_x,n})\right]$. Following the same procedure for the elements of $\mathbf{W}^b$ we obtain the relation:

$$\frac{\partial \mathbf{x}_{n+1}}{\partial W_{ij}^b} = \mathbf{\Phi}'(\mathbf{z}_n)\left(\mathbf{I}_{ij}\mathbf{u}_n + \mathbf{W}^a \frac{\partial \mathbf{x}_n}{\partial W_{ij}^b}\right) \qquad \begin{matrix} i = 1, \ldots, n_x \\ j = 1, \ldots, m \end{matrix} \tag{14}$$

At this point, assembling the different blocks (13) and (14), we can define:

$$\mathbf{U}_n = \left[\mathbf{\Phi}'(\mathbf{z}_n)\mathbf{I}_{ij}\mathbf{x}_n \quad \mathbf{\Phi}'(\mathbf{z}_n)\mathbf{I}_{ij}\mathbf{u}_n\right] \tag{15}$$

and

$$\hat{\mathbf{A}}_n = \mathbf{\Phi}'(\mathbf{z}_n)\mathbf{W}^a \tag{16}$$

ending with a time dependent Jacobian matrix governed by the following dynamic model:

$$\mathbf{\Lambda}_{n+1} = \hat{\mathbf{A}}_n \mathbf{\Lambda}_n + \mathbf{U}_n \tag{17}$$

Eventually, the coupling of Eq. (1) and Eq. (17) fully defines the nonlinear state dynamics of a DTRNN:

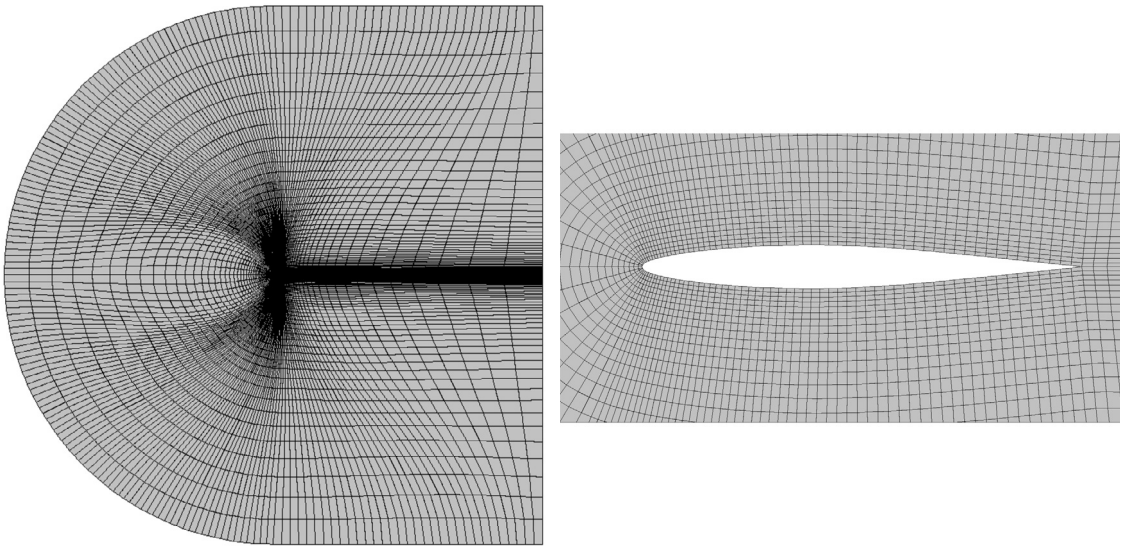
$$\begin{cases} \mathbf{x}_{n+1} & = \mathbf{\Phi}\left(\mathbf{W}^a \mathbf{x}_n + \mathbf{W}^b \mathbf{u}_n\right) \\ \mathbf{\Lambda}_{n+1} & = \hat{\mathbf{A}}_n \mathbf{\Lambda}_n + \mathbf{U}_n \\ \mathbf{y}_n & = \mathbf{W}^c \mathbf{x}_n \\ \mathbf{J}_n & = -\mathbf{W}^c \mathbf{\Lambda}_n \end{cases} \tag{18}$$

Thanks to the structure of $\mathbf{W}^c$, highlighted by Eq. (2), it is possible to assign the initial conditions required by Eq. (18) as the first value of the outputs of the related high fidelity simulation. In this way, the neural network acquires a basic physical connotation, where the value of the first state variables is directly the value of the aerodynamic loads. Therefore, prior knowledge is introduced to the system represented by Eq. (18) through its initial conditions:

$$\mathbf{x}_0 = \left(\hat{\mathbf{y}}_0^T \quad \mathbf{0}^T\right)^T \tag{19}$$

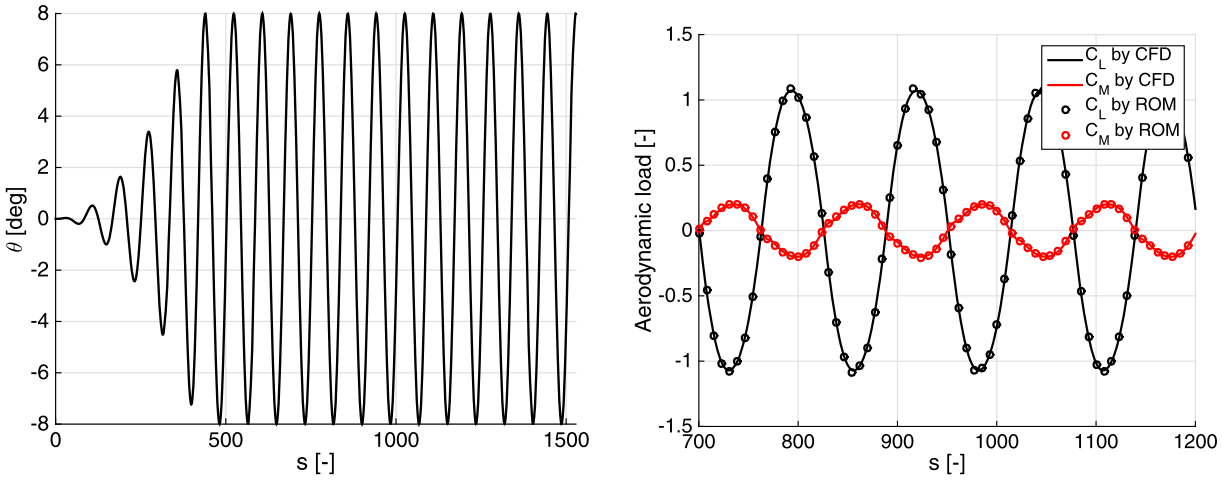
where the size of the null vector is equal to $n_x - p$. On the other hand, the initial condition of the state Jacobian matrix is taken as a null matrix, meaning that the initial synaptic weights of the network reside at a stationary point of $\mathbb{R}^{n_\theta}$:

$$\mathbf{\Lambda}_0 = \mathbf{0} \tag{20}$$



(a) C-type topology mesh (b) Mesh close-up

Fig. 2. Computational mesh used in the present Euler-based calculations.



(a) Harmonic training input (b) Related aerodynamic loads

Fig. 3. Training signal and related results. The forcing reduced frequency is $k = 0.15$.

3. Results

3.1. Pitching airfoil in transonic flow

The often used NACA 64A010 airfoil [9,16], pitching around the 20% of its chord is considered here for a first convergence analysis of the proposed DTRNN scheme. The working condition is an air flow at $M_\infty = 0.8$.

As shown in Fig. 2, the adopted two dimensional mesh is of a C-type with 12 200 cells, resulting in a total of 48 800 unknowns, since density, momentum and total energy are unknown in each cell. The present test is a simple aerodynamic load identification, leading to a single input (pitch angle), multiple outputs (lift and moment coefficients) problem.

Following [16], the network is trained through the imposition of a harmonic pitching motion of period T, selected case by case, represented by:

$$\alpha(t) = A(t)\sin(\omega t) \tag{21}$$

whose amplitude is:

$$A(t) = \begin{cases} \left(\dfrac{t}{nT}\right)^r A_0 & t < nT \\ A_0 & t \geq nT \end{cases} \tag{22}$$

with $n = 5$, $r = 3$ and $A_0 = 8$ deg. The training is tackled over two input histories. At first a signal with reduced frequency $k = \omega c/V_\infty = 0.15$ is presented to the DTRNN, followed by one with $k = 0.05$, aiming at improving the generalization properties of the network. An example of such signals is given in Fig. 3(a). These signals are expressed as function of the dimensionless time $s = V_\infty t/b$, where b is the airfoil semi-chord.

A convergence analysis is carried out here to compare the accuracy and the computational time required by the proposed training procedure. Let us call A the DTRNN characterized by $n_x = 8$ and thus $n_\theta = 72$, where again n_x represents the number of neurons and n_θ the total number of parameters describing the network, B the network with $n_x = 10$, $n_\theta = 110$ and C the network with $n_x = 12$, $n_\theta = 156$.

The training results are resumed in Table 1, comparing the outcomes related to the evaluation of the Jacobian matrix in the LM

Table 1
Convergence test of the three different DTRNNs considered in this case.

Model	Jacobian matrix computational method	Computational time [min]	Final cost function value
DTRNN A	Finite differences	25	1.24
	Analytical	6	0.11
DTRNN B	Finite differences	38	0.25
	Analytical	10	0.032
DTRNN C	Finite differences	62	0.055
	Analytical	24	0.00069

iterations, either through finite differences or analytically, as detailed in Section 2.3.

The comparison between the two methods is considered here to assess if the knowledge of the analytical Jacobian matrix could bring improvements to the proposed training procedure in terms of precision and convergence to the reference results when compared to a simpler calculation of the same quantity by means of a finite difference formula. From Table 1 can be seen that, compared to its analytical counterpart, the simpler finite difference evaluation of the Jacobian matrix of the LM method is rather inefficient. Moreover, comparing the results achieved by networks A, B and C, it appears that B shows the best trade-off between accuracy and required computational time.

An example of the training outcomes is shown in Fig. 3(b). From this figure can be noted that the present training signal well excites the system nonlinearities. In fact, the behavior of the aerodynamic moment is clearly nonlinear, because its shape is different from the one of the input signal. The coefficient of lift instead is rather linear with respect to the harmonic input, because the related time history is similar in shape to the input's one. This result is common in aerodynamic problems modeled through Euler flows [15,16,36].

All of the computations are carried out with a single processor, Intel® Core™ 2 Duo CPU, 2.93 GHz of maximum frequency and 4 GB of RAM memory. The time required for the generation of the training signal by AeroFoam is 3 h 10 min for the first signal and 3 h 50 min for the second, adopting a time step $\Delta t = 10^{-3}$ s, equal to $\Delta s = 0.544$ dimensionless time steps.

The maximum number of generations admitted for the GA phase is 200, starting from random synaptic weights, while the LM method is allowed to reach 300 iterations, with a cost function threshold value fixed at 10^{-4}, assessed to be small enough to achieve meaningful identification results. To weigh the fitting error uniformly, both input and output are normalized with respect to their maximum value.

For validating the present results, three sinusoidal pitching motions are considered:

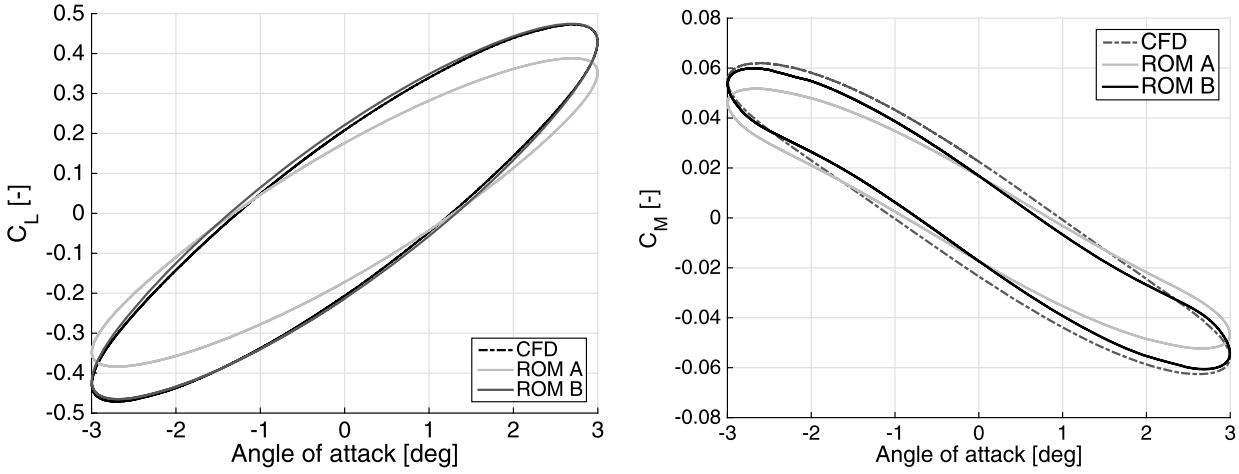
$$\alpha(t) = A_0 \sin(\omega t) \tag{23}$$

The first two cases are described by amplitude $A_0 = 3$ deg and reduced frequency $k = 0.12$ and 0.24 respectively. The related signals frequencies are remarkably different from those used to train the DTRNN, and for such a reason this can be considered a good generalization test for the present method. Nonetheless, the signals amplitude is rather small in this case, thus a weakly nonlinear response is expected. For this reason, a third test case is considered with $A_0 = 7$ deg and $k = 0.1$. Such a case is supposed to be highly nonlinear because of the large motion amplitude involved and it is used to evaluate the ability of the DTRNN in capturing the essential nonlinear characteristics of the response.

The related results are shown in Figs. 4 and 5. A hysteresis loop, typical of unsteady aerodynamic responses [37], is obtained. As can be noticed from both the figures, all the DTRNNs replicate the reference results accurately. Nonetheless, it is clear that increasing the number of neurons, the precision increases. For example, passing from DTRNN A to B permits to reduce the differences with respect to the outcomes computed by the CFD code. The results obtained with DTRNN C are not shown here because they are almost identical to those obtained by DTRNN B. It is also interesting to see how the loads predicted by the DTRNNs are far more accurate for the lift rather than for the moment coefficient. This could be due to the strong nonlinear behavior of this latter, which may lead to a difficult identification of the relation between the airfoil motion and this aerodynamic load.

The results related to the third test case are presented in Fig. 6. Again, the DTRNNs are able to replicate the reference results with good fidelity, showing an improved precision as the number of neurons is increased.

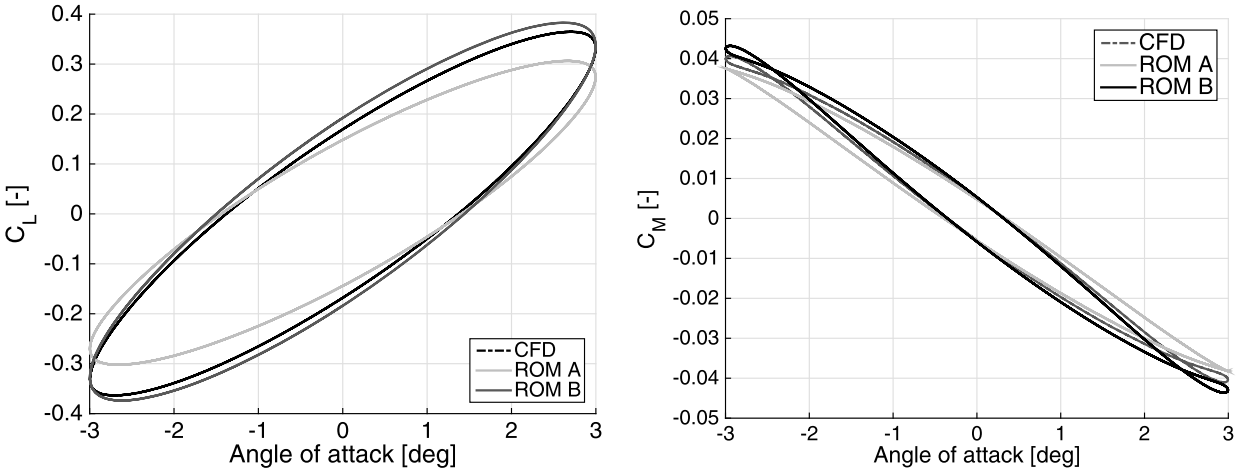
To highlight the stronger nonlinear behavior of this last verification case with respect to the others, a comparison of the moment coefficient fast Fourier transform (FFT) is proposed in Fig. 7. As evident, Fig. 7(b), relative to the third verification signal, shows six detectable peaks at frequencies multiple of the fundamental one, being this a peculiarity of nonlinear systems responses [24]. On the other hand, in Fig. 7(a), relative to the first case, only two peaks are detected. It is then clear that an oscillation of 3 deg at $k = 0.12$ is only weakly nonlinear when compared to an oscillation of 7 deg at $k = 0.1$.



(a) Lift coefficient versus airfoil rotation

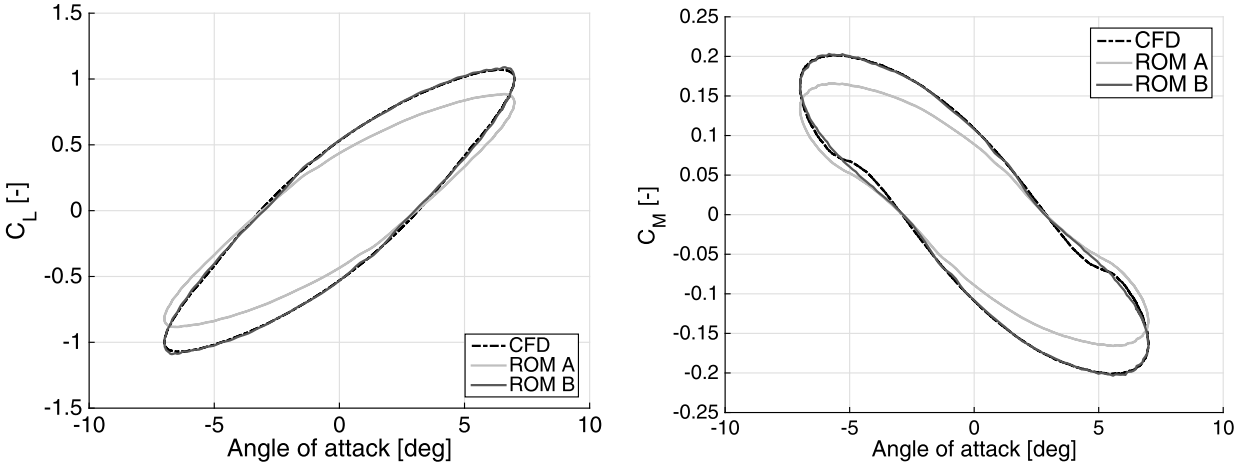
(b) Moment coefficient versus airfoil rotation

Fig. 4. First validation test with $A_0 = 3$ deg at $k = 0.12$.



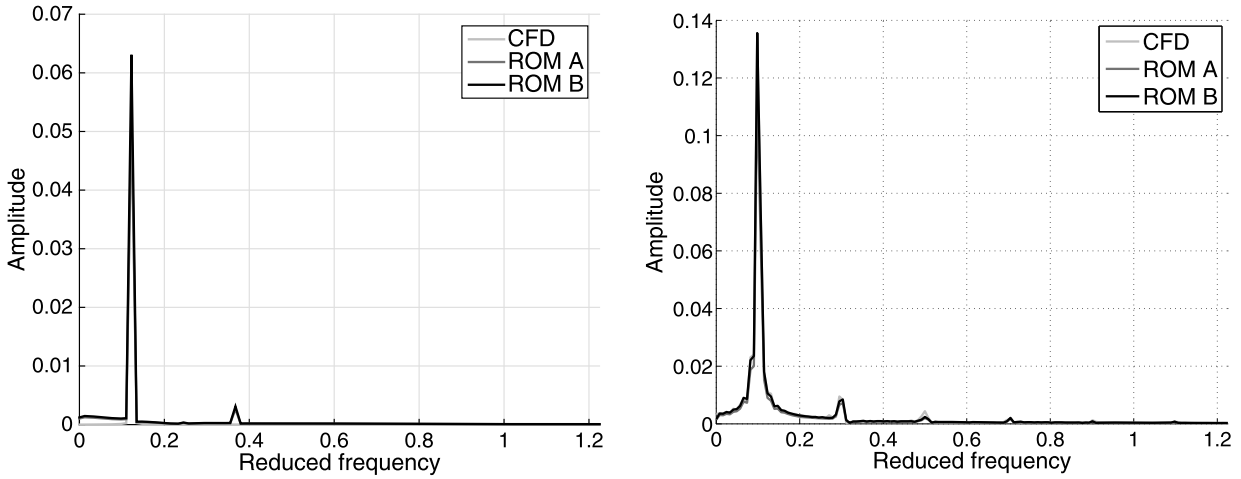
(a) Lift coefficient versus airfoil rotation (b) Moment coefficient versus airfoil rotation

Fig. 5. Second validation test with $A_0 = 3$ deg at $k = 0.24$.



(a) Lift coefficient versus airfoil rotation (b) Moment coefficient versus airfoil rotation

Fig. 6. Third validation test with $A_0 = 7$ deg at $k = 0.10$.



(a) Moment coefficient FFT – first validation test (b) Moment coefficient FFT – third validation test

Fig. 7. Moment coefficient FFT computed using the forcing signals of the first and the third validation cases.

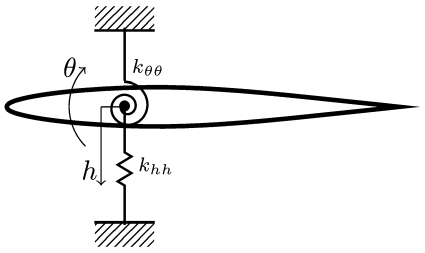


Fig. 8. Pitching and plunging typical section.

Nevertheless, the DTRNN-based method is able to capture all these peaks in both cases, proving to be an efficient tool in the identification of nonlinear and unsteady aerodynamic loads.

The computational time required by AeroFoam for such a simulation is 3 h and 10 min, with the same time step of the training signal. The same simulation carried out using the simpler DTRNN based ROM requires only 1 s. The cost function defined by Eq. (6) is reduced to a value close to $F = 0.001$. Thus, even with the very low order adopted, i.e. 10 neurons, the accuracy of the reduced order model remains high. As mentioned before, because the DTRNN B shows the best trade-off between training time required and final accuracy, it will be the one considered in the next applications.

Even though this simple example application does not show the full potential of the DTRNN as reduced order model, because the computational cost comparison is made over only one simulation, it is demonstrated how the network is able to generalize input signals that were never seen during the training phase, nonetheless reproducing reliable results.

3.2. Two degree of freedoms typical section

The DTRNN is applied here to a plunging (h, positive downward) and pitching (θ, positive clockwise) typical section with the same airfoil and flight condition as for the previous applications. A basic layout of such a mechanical system is shown in Fig. 8. The related non-dimensional equations of motion are given by:

$$\begin{bmatrix} 1 & x_\theta \\ x_\theta & r_\theta^2 \end{bmatrix} \begin{Bmatrix} h/b \\ \theta \end{Bmatrix}'' + \begin{bmatrix} (\omega_h/\omega_\theta)^2 & 0 \\ 0 & r_\theta^2 \end{bmatrix} \begin{Bmatrix} h/b \\ \theta \end{Bmatrix} = \mathbf{F}_{\text{aero}} + \begin{bmatrix} 0 \\ r_\theta^2 \end{bmatrix} \theta_0,$$

$$\mathbf{F}_{\text{aero}} = \frac{(V^*)^2}{\pi} \begin{Bmatrix} C_L(h/b, \theta, h'/b, \theta') \\ 2\, C_{M_{EA}}(h/b, \theta, h'/b, \theta') \end{Bmatrix} \tag{24}$$

Table 2
Data of the typical section case.

x_θ	r_θ^2	ω_h/ω_θ	μ
0.25	0.75	0.5	75

where x_θ and r_θ are the dimensionless static unbalance and moment of inertia, respectively, ω_h and ω_θ the uncoupled natural circular frequencies of the mechanical system. The dynamic model is written in terms of the non-dimensional time $\tau = \omega_\theta t$, and the bifurcation parameter is represented here by the reduced velocity V^*, defined as:

$$V^* = \frac{V_\infty}{\omega_\theta\, b\, \sqrt{\mu}} \tag{25}$$

being b the semi-chord and μ the fluid to mass ratio. The parameter θ_0 in Eq. (24) is a non-null pre-twist angle that will be used to test the model performance outside its training set, computed considering $\theta_0 = 0$ deg. Of course the effect of such a parameter will be to make the resulting aeroelastic LCOs no more symmetric.

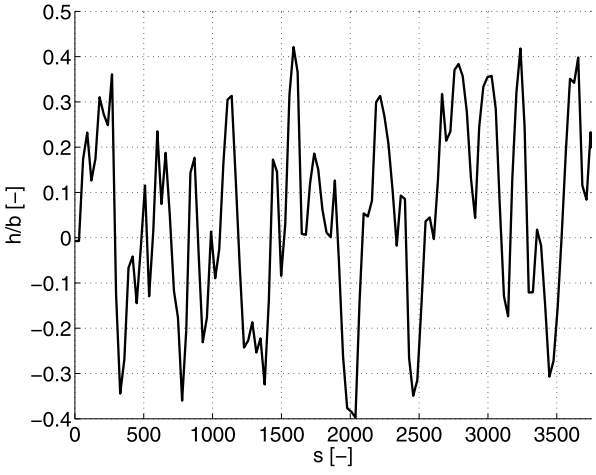
The model can be written in the following compact form:

$$\mathbf{M}\mathbf{u}'' + \mathbf{K}\mathbf{u} = \mathbf{f} + \mathbf{K}_0\theta_0 \tag{26}$$

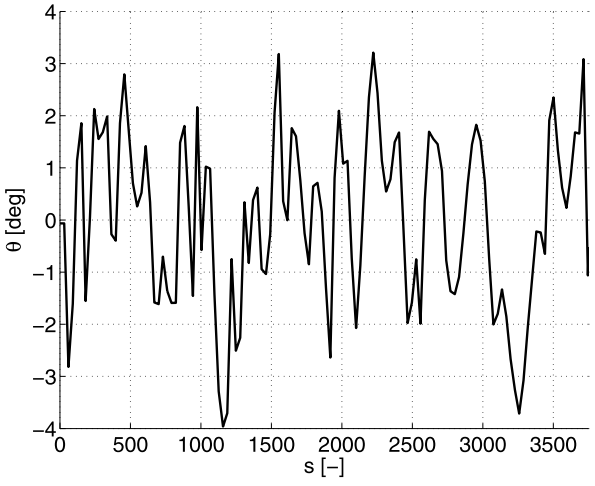
while the parameters of the case considered here are summarized in Table 2.

Differently from what presented in [15,16], the training signal used in this case is designed without considering the actual aeroelastic coupling. Its generation comes from a purely aerodynamic simulation with imposed boundary conditions, which should be able to catch all of the motion amplitudes and frequencies of interest. The designed training signal excites a range of reduced frequencies up to $k = 0.6$. White noise-like signals, as the ones shown in Fig. 9 are selected because they seem to possess good properties for such a nonlinear identification [17]. The FFT of these signals is depicted in Fig. 10, where it is evident how the reduced frequency range $k = 0.05$–0.6 is well excited, while out of it the signal strength is rather low. It may also be noted that the reduced frequency $k = 0$ is not excited because no steady state components of the signal are used in the training.

Because the discrete time framework presented in Section 2.3 is limited by the fact that the adopted time step is fixed once for all when the training phase has been accomplished, this must be carefully chosen small enough to represent any response of interest in the analyses that follow the training. A physical time step of

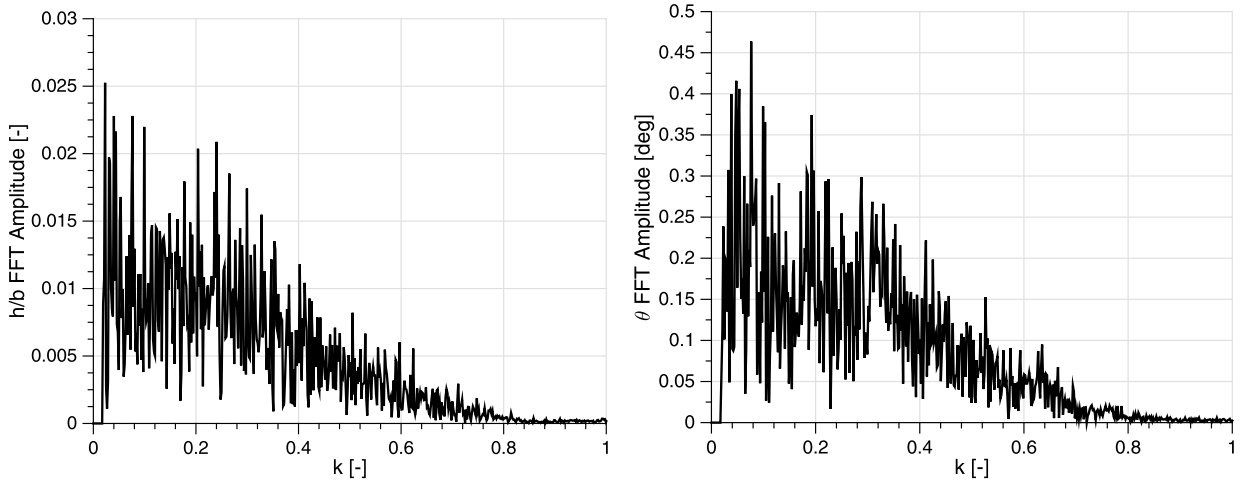


(a) Airfoil plunge motion



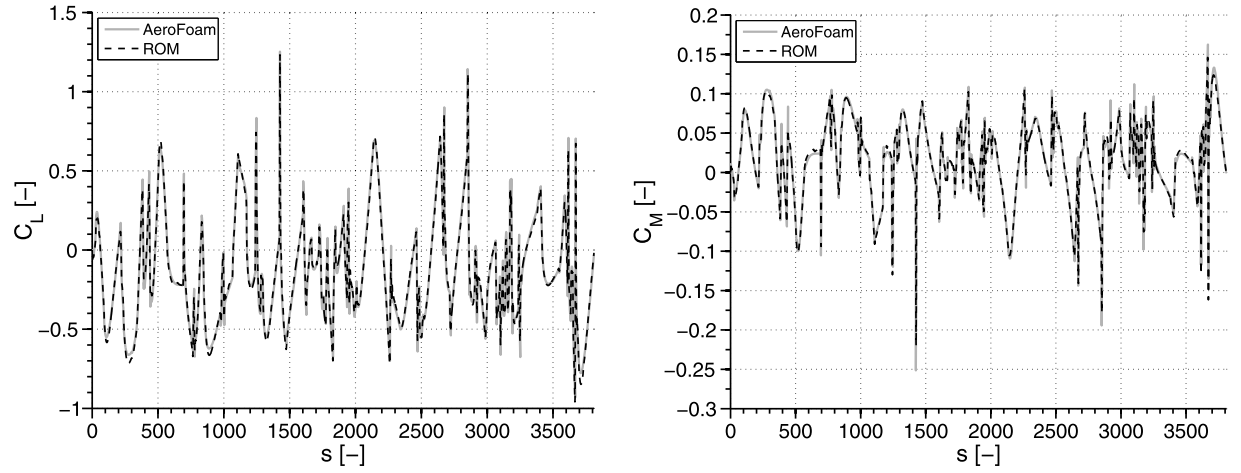
(b) Airfoil pitch motion

Fig. 9. Training signal used in the two degree-of-freedom typical section case.



(a) Plunge motion FFT (b) Pitch motion FFT

Fig. 10. Training signal FFT used in the two degree-of-freedom typical section case.



(a) Lift coefficient (b) Moment coefficient

Fig. 11. Aerodynamic loads identification resulting from the training phase.

$\Delta t = 2 \cdot 10^{-3}$ s (or $\Delta s = 1.088$ in dimensionless time) is set in the simulation of the training signal, with this value considered to be adequate for the series of analyses that will be performed in this section.

The solver AeroFoam generates the related training data in about 20 h. The training results, obtained with a network characterized by $n_x = 10$, are shown in Fig. 11. The training time required is 2 h 50 min. Such a large increase with respect to the previous test case is due to the longer training horizon and to the wide frequency spectrum excited. The cost function defined by Eq. (6) is brought down to $F = 0.08$.

It is then possible to couple the DTRNN-based aerodynamic model to the structural system, so obtaining a nonlinear aeroelastic model, which is described by:

$$\begin{cases} \mathbf{M}\ddot{\mathbf{u}} + \mathbf{K}\mathbf{u} = \mathbf{f} + \mathbf{K}_0\theta_0 \\ \mathbf{f}_n = \mathbf{B}\mathbf{W}^c\mathbf{x}_n \\ \mathbf{x}_{n+1} = \boldsymbol{\Phi}\left(\mathbf{W}^a\mathbf{x}_n + \mathbf{W}^b\mathbf{u}_n\right) \end{cases} \tag{27}$$

with $\mathbf{B} = \dfrac{(V^*)^2}{\pi}\mathrm{Diag}\,(1,\ 1/2)$. After assigning an initial condition, the dynamic system is integrated forward in time. In this case, the reduced velocity V^* is the bifurcation parameter, which determines the system stability properties.

Such a time integration can be carried out with explicit or implicit schemes. While explicit methods are usually the first choice thanks to their simplicity of implementation, they are constrained to limit their time steps because of numerical stability issues. Implicit methods permit larger integration time steps, albeit at the cost of solving a nonlinear algebraic problem at each step. This fact does not cause any major drawback if the system has a relatively small size or if it presents a simple analytical expression.

When integrating forward in time the reduced order model represented by the present structural model and the DTRNN, this is indeed the case. Therefore an implicit time-marching scheme, like the one developed in [38], is adopted here for the integration of the continuous part of Eq. (27), while the DTRNN is simply updated through its recursive framework. The application of such a method results in a two-step, L-stable scheme of the type:

$$\mathbf{z}_{n+1} = a_0\mathbf{z}_n + a_{-1}\mathbf{z}_{n-1} + b_1\mathbf{R}\,(\mathbf{z}_{n+1}) + b_0\mathbf{R}\,(\mathbf{z}_n) + b_{-1}\mathbf{R}\,(\mathbf{z}_{n-1}) \tag{28}$$

where the subscript n characterizes the solution $\mathbf{z}$ at the n-th time step. The term $\mathbf{R}\,(\mathbf{z})$ represents the right hand side of Eq. (27). As shown in [38], the coefficients a_i and b_i will depend on the adopted time discretization and the desired level of numerical dissipation.

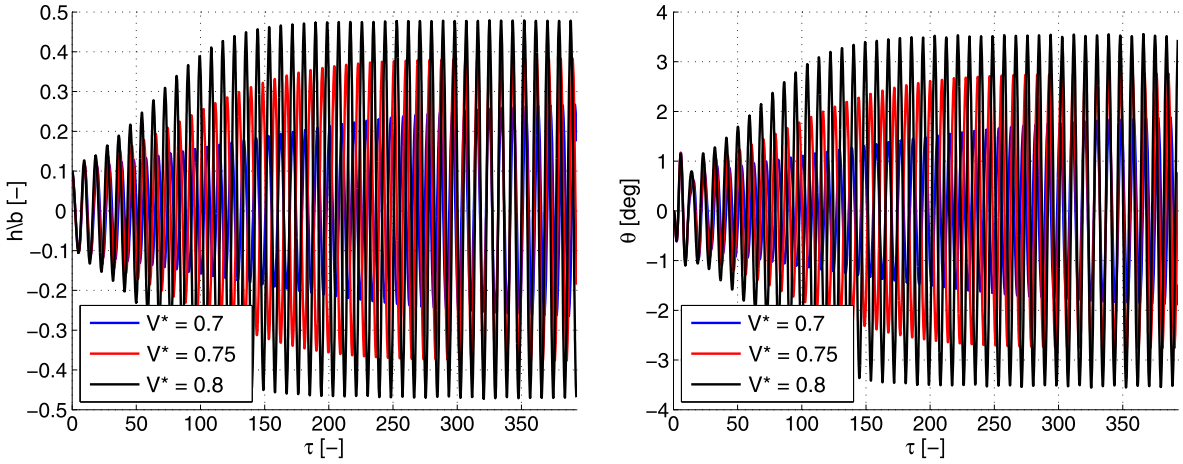
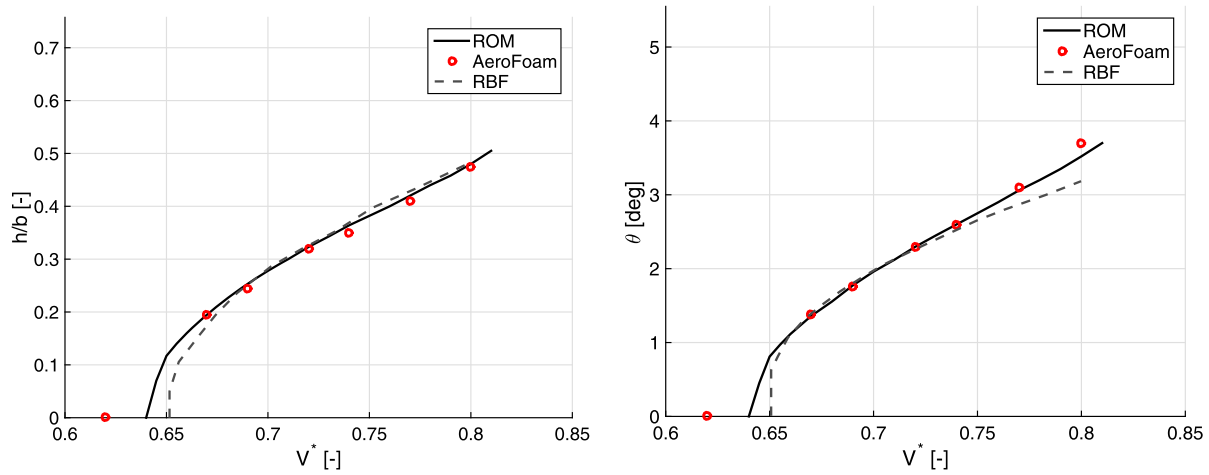


Fig. 12. Sample of LCOs obtained with the DTRNN-based model at different flight speeds. (For interpretation of the colors in this figure legend, the reader is referred to the web version of this article.)



(a) Plunge trend (b) Pitch trend

Fig. 13. Comparison of LCO amplitude trends.

Fig. 12 shows a few LCO solutions obtained at three different reduced velocities using such an integration method. From this figure, it can be noted that as the reduced velocity is increased, the LCOs amplitude increases as well. Physically, in transonic flows mainly influenced by compressibility effects, LCOs are brought into the system due to the large motion of strong shock waves interacting with the oscillating airfoil. Such an increase of amplitude is therefore related to the larger distance swept by the shock wave during one LCO cycle.

In addition, it is now possible to compute the LCO amplitude trends for varying reduced velocities, of the types shown in Fig. 13, where the results are compared with those obtained by AeroFoam and the method developed in [16], that employs an approach based on radial basis function networks, labeled as RBF. The same kind of comparison is shown in Fig. 14 about the LCO frequency trends. As can be noticed from these figures, the ROM-based aeroelastic model predicts LCOs very similar to those computed by the high fidelity CFD code. Having a further look at Fig. 13, it is evident how the LCOs are a strongly nonlinear phenomenon, because they involve plunge motions of about 25% the airfoil chord and rotations in the order of 4 degrees. Such results demonstrate that the proposed ROM can evaluate correctly these peculiar nonlinear responses, even on the base of a training that does not consider a harmonic signal as input. Clearly, the main advantage of such a model is the huge saving in computational time compared to the

Table 3
CFD-based simulations, computational time required.

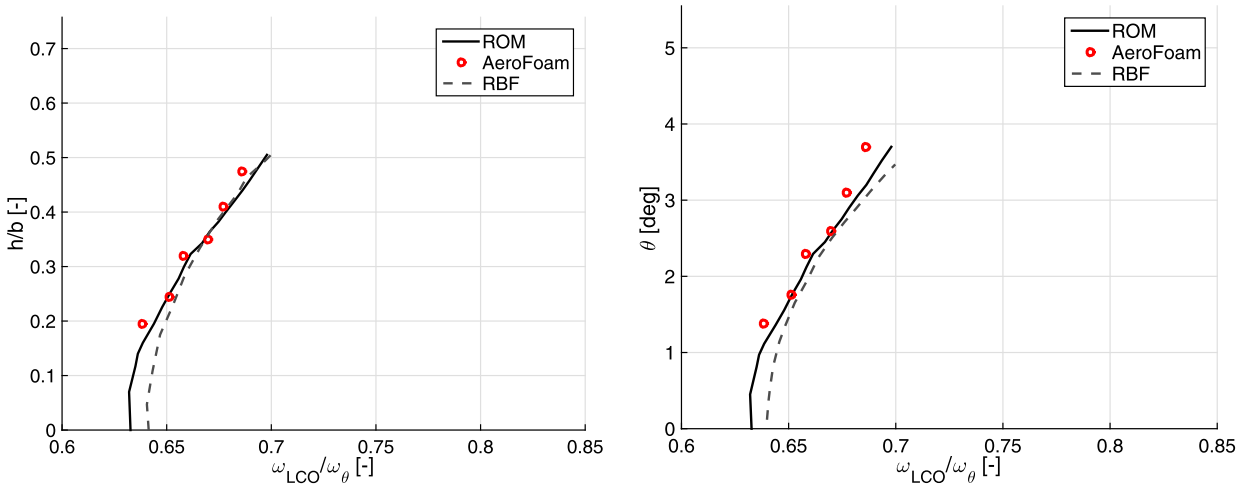
Time-accurate CFD simulations (7 points)	**244 h 06 min**

Table 4
ROM-based simulations, computational time required.

Generation of the training input–output data pair by AeroFoam	20 h 00 min
Training time	2 h 50 min
Envelope simulation time (20 points)	5 min
Total time required	**22 h 55 min**

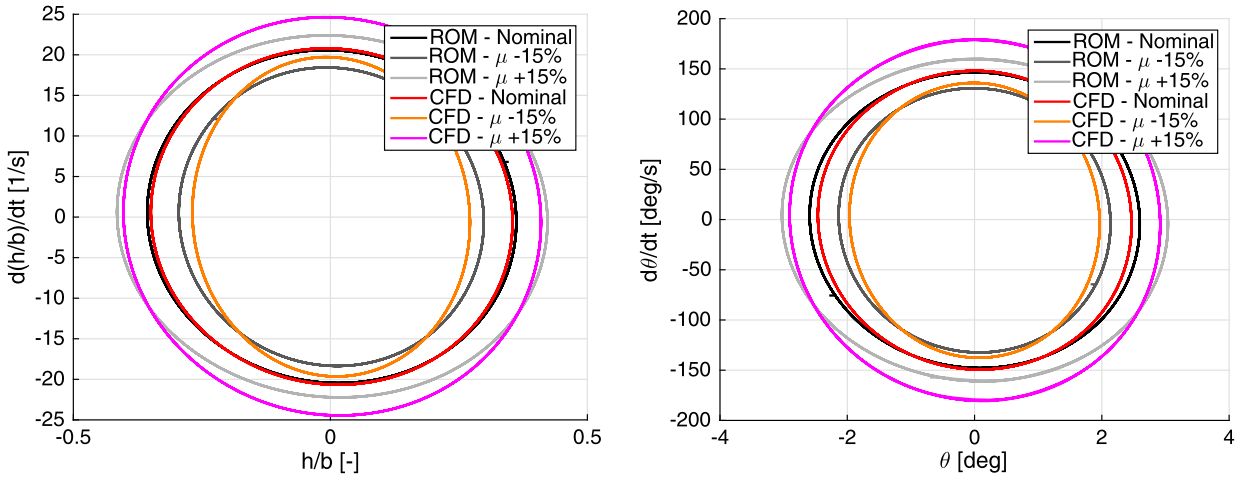
LCO trends obtained by CFD-based calculations. The required computational efforts are shown in Tables 3 and 4 respectively. From such a comparison, it is evident that the DTRNN permits to reduce the computational time of the LCO trends by a factor of 10, while maintaining a good accuracy.

Having proven the convergence of the DTRNN-based aeroelastic system to the results obtained by high fidelity simulations, it is now time to test the performance of the system when some key parameters are changed. In the following analyses, perturbations of ±15% are applied to μ, x_θ and ω_r, with their meaning resumed by Eq. (24). All the results are presented at the reduced velocity $V^* = 0.74$, and the comparison between the DTRNN- and CFD-based responses is provided in the phase space, to highlight



(a) Plunge trend (b) Pitch trend

Fig. 14. Comparison of LCO frequency trends.



(a) Plunge LCOs (b) Pitch LCOs

Fig. 15. Comparison of LCO responses with different values of fluid-to-mass ratio μ. (For interpretation of the colors in this figure legend, the reader is referred to the web version of this article.)

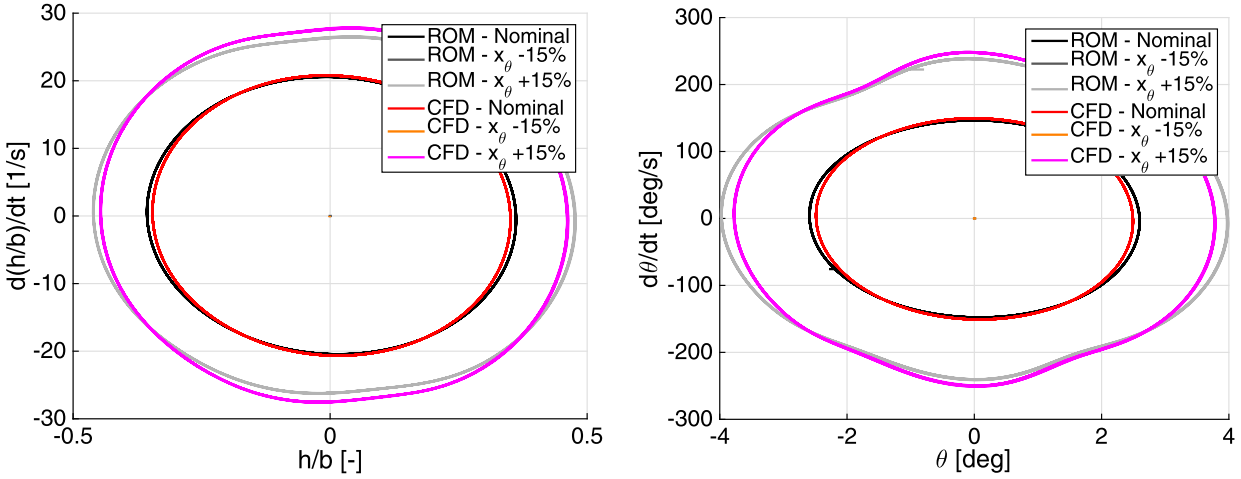
the variations of the LCO shape induced by the changes of such variables. In Fig. 15 are depicted the responses obtained with various values of the fluid-to-mass ratio μ. As can be noticed, such variations do not introduce large modifications in the LCOs shape, which remains almost circular for all the values of μ considered. Also in this perturbed case, the DTRNN is able to reproduce accurate responses compared with CFD calculations. Giving a further look at Fig. 15, it seems that the reduced order model is more precise in predicting the LCO amplitude rather than the LCO velocity, even if these differences may be considered negligible, since the related relative error is always smaller or equal than 10%.

Resulting from similar analyses, in Fig. 16 are shown the responses obtained with various values of the dimensionless static unbalance x_θ. Differently from the previous case, these variations introduce visible modifications in the LCOs shape, especially the one related to the pitch degree of freedom. Furthermore, when x_θ is reduced by the 15% of its original value, the system is stable and does not show any LCO behavior at this flight speed. Again, the DTRNN is still able to reproduce these stability changes when compared with the reference CFD results.

In Fig. 17 are presented the responses obtained with various values of the frequency ratio ω_r. Once again, these variations modify substantially the LCOs shape, and when ω_r is reduced by the 15% of its original value, the system results stable at this flight speed.

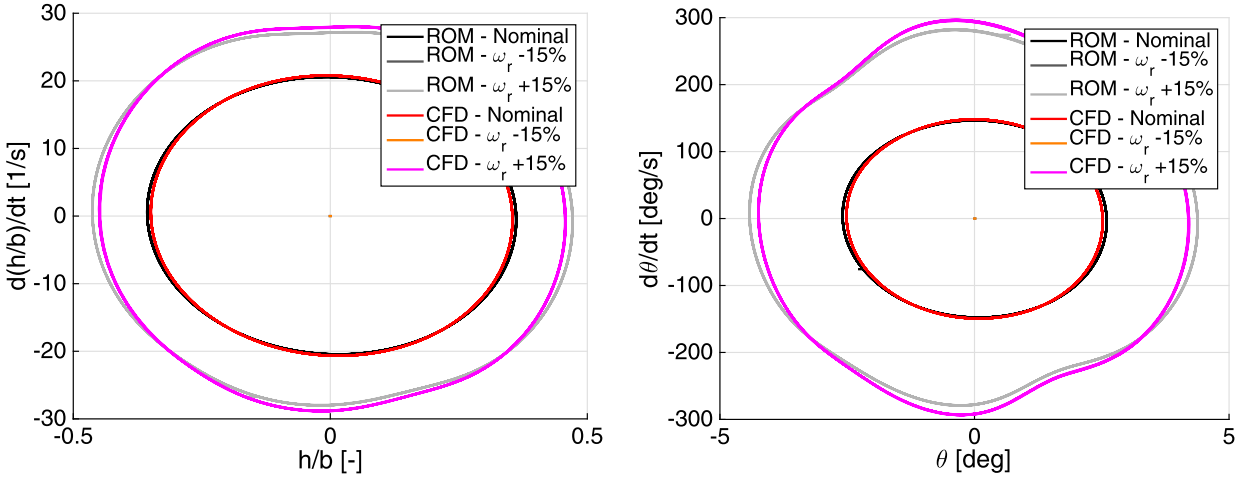
Finally, the influence of a non-null pre-twist angle θ_0, introduced in Eq. (24), is shown. Responses with $\theta_0 = \pm 1$ deg are considered, while all the other variables are kept at their nominal value. Similarly to variations of μ, the present perturbations do not introduce large modifications to the LCOs shape, which remains almost circular for the values considered. The reduced order model is robust enough to maintain a good accuracy in this case also, even if the perturbation introduced by θ_0 leads to responses not included in the training set, since they present a non-null mean value, as depicted in Fig. 18. Notice that the variables on the vertical axes are now $\Delta h/b$ and $\Delta\theta$, computed as $[\max(\text{LCO}) - \min(\text{LCO})]/2$, because the LCOs are no longer symmetric in this case.

Obviously, increasing the value of the pre-twist, for example at 2 degrees, the response of the aeroelastic system changes its nature and the DTRNN is no longer able to follow the reference LCO trends computed by the CFD. This fact is well shown in Fig. 19, where it is clear that the CFD results present a very different trend with respect to the one computed with a null pre-twist, while the DTRNN still replicates an envelope similar to those shown in Fig. 13. This misleading behavior could be fixed considering train-



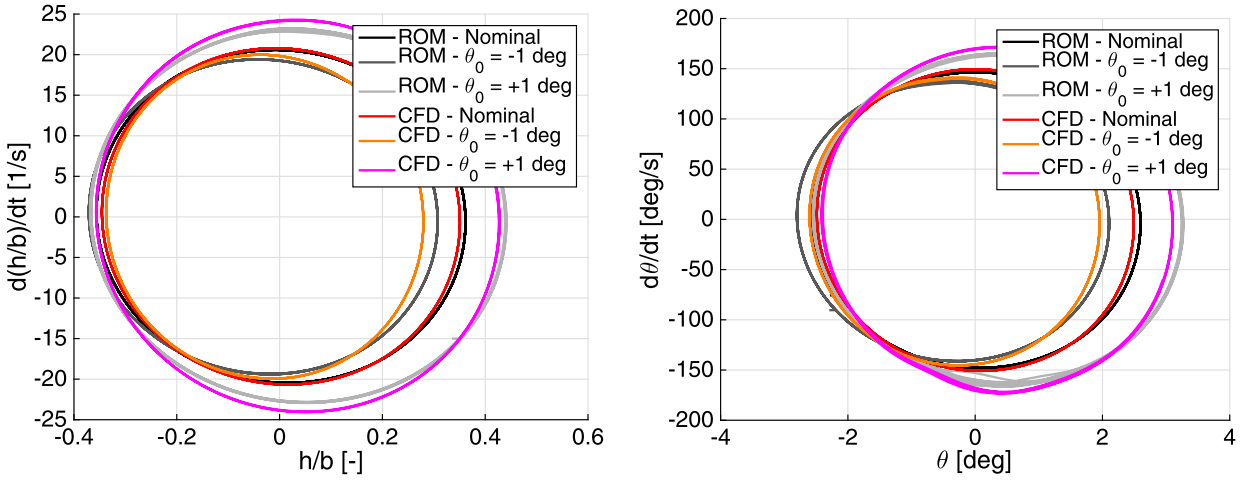
(a) Plunge LCOs (b) Pitch LCOs

Fig. 16. Comparison of LCO responses with different values of the dimensionless static unbalance x_θ. (For interpretation of the colors in this figure legend, the reader is referred to the web version of this article.)



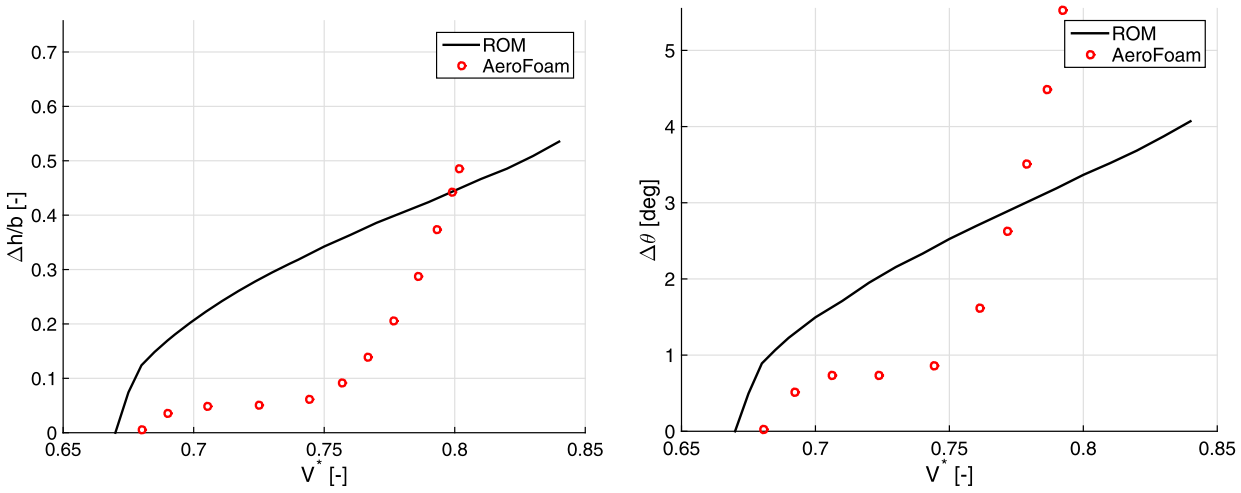
(a) Plunge LCOs (b) Pitch LCOs

Fig. 17. Comparison of LCO responses with different values of the frequency ratio ω_r. (For interpretation of the colors in this figure legend, the reader is referred to the web version of this article.)



(a) Plunge LCOs (b) Pitch LCOs

Fig. 18. Comparison of LCO responses with different values of the pre-twist angle θ_0. (For interpretation of the colors in this figure legend, the reader is referred to the web version of this article.)



(a) Plunge trend (b) Pitch trend

Fig. 19. Comparison of LCO amplitude trends with a pre-twist angle $\theta_0 = 2$ deg.

ing signals with a non-null mean value. Such a solution is currently under investigation by the authors, but it goes beyond the scope of the present work.

The present result demonstrates that there is still a lot of work to do if an 'infinitely' robust ROM is desired. Nonetheless, it has been proven that the present DTRNN-based ROM is robust to key parameters variations greater than 10% of their nominal value, and this could be considered a good level of robustness already. If more robust reduced order models are needed, then techniques belonging to the field of uncertainty quantification should be employed, and this option is currently under consideration by the authors.

4. Concluding remarks

This effort has detailed the development of a neural network-based aerodynamic reduced order model, trained on input–output data pairs generated by a generic CFD code, followed by its use in the calculation of aerodynamic harmonic responses and aeroelastic limit cycles. The related performances have shown a level of accuracy dependent on the type of nonlinear functions considered in the model, the number of neurons employed and the signal amplitude and frequency used in the training of the neural network. The reduced order model has proven to be quite robust in the face of variations of the reduced velocity and other key parameters. For this reason, a model built only on the results of pure aerodynamic simulations can be employed in any aeroelastic system at various reduced velocities and for a large set of model parameters, with the only restriction of maintaining the Mach number fixed.

The efficiency of the presented reduced order model is therefore quite appealing, and thanks to its limited computational effort compared to full order analyses, in the future it might make possible the inclusion of nonlinear dynamics phenomena even in the first stages of an aircraft design, maintaining an adequate level of accuracy. Nevertheless, it is believed that to fully verify the strength and weaknesses of this reduced order model methodology, there remains the need of focusing on more complex and realistic applications, e.g. flexible wings with control surfaces, deformable free-flying aircraft, and finally testing the robustness of the reduced order model in the face of Mach number variations.

Conflict of interest statement

None declared.

References

[1] M. Ghoreyshi, D. Vallespin, A. Da Ronch, K.J. Badcock, J. Vos, S. Hitzel, Simulation of aircraft manoeuvres based on computational fluid dynamics, in: AIAA Atmospheric Flight Mechanics Conference, 2010.
[2] A. Schütte, R.M. Cummings, T. Loeser, An integrated computational/experimental approach to X-31 stability and control estimation, Aerosp. Sci. Technol. 20 (1) (2012) 2–11.
[3] G. Romanelli, M. Castellani, P. Mantegazza, S. Ricci, Coupled CSD/CFD nonlinear aeroelastic trim of free-flying flexible aircraft, in: 53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference, American Institute of Aeronautics and Astronautics, 2012.
[4] A. Beckert, Coupling fluid (CFD) and structural (FE) models using finite interpolation elements, Aerosp. Sci. Technol. 4 (1) (2000) 13–22.
[5] S.A. Morton, P.S. Beran, Hopf-bifurcation analysis of airfoil flutter at transonic speed, J. Aircr. 36 (1999) 421–429.
[6] D.J. Lucia, P.S. Beran, W.A. Silva, Reduced-order modeling: new approaches for computational physics, Prog. Aerosp. Sci. 40 (2004) 51–117.
[7] W.H. Schilders, H.A. van der Vorst, J. Rommes, Model Order Reduction: Theory, Research Aspects and Applications, Springer, 2008.
[8] L. Sirovich, Turbulence and the dynamic of coherent structures, part *I*: coherent structures, Q. Appl. Math. 45 (1987) 561–571.
[9] J.P. Thomas, E.H. Dowell, K.C. Hall, Nonlinear inviscid aerodynamic effects on transonic divergence, flutter, and limit-cycle oscillations, AIAA J. 40 (2002) 638–646.
[10] G. Chen, Y. Li, G. Yan, Nonlinear POD reduced order model for limit cycle oscillation prediction, Sci. China, Phys. Mech. Astron. 53 (7) (2010) 1325–1332.
[11] K.C. Hall, J.P. Thomas, W.S. Clark, Computation of unsteady nonlinear flows in cascades using a harmonic balance technique, AIAA J. 40 (2002) 879–886.
[12] M. Spiker, J.P. Thomas, K.C. Hall, R. Kielb, E.H. Dowell, Modeling cylinder flow vortex shedding with enforced motion using a harmonic balance approach, in: 47th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, 2006, AIAA paper 2006-1965.
[13] S. Timme, S. Marques, K.J. Badcock, Transonic aeroelastic stability analysis using a Kriging-based Schur complement formulation, AIAA J. 49 (2011) 1202–1213.
[14] S. Timme, K.J. Badcock, Transonic aeroelastic instability searches using sampling and aerodynamic model hierarchy, AIAA J. 49 (2011) 1191–1201.
[15] W. Zhang, B. Wang, Z. Ye, High efficient numerical method for limit cycle flutter analysis based on nonlinear aerodynamic reduced-order-model, in: 51st AIAA, ASME, ASCE, AHS, ASC Structures, Structural Dynamics and Material Conference, Orlando, FL, 2010.
[16] W. Yao, M. Liou, Reduced-order modeling for flutter/LCO using recurrent artificial neural network, in: 12th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference and 14th AIAA/ISSM, Indianapolis, IN, 2012.
[17] G. Chen, Y. Zuo, J. Sun, Y. Li, Support-vector-machine-based reduced-order model for limit cycle oscillation prediction of nonlinear aeroelastic system, Math. Probl. Eng. 2012 (2012).
[18] O. Nelles, Nonlinear System Identification, Springer, 2001.
[19] D.W. Jordan, P. Smith, Nonlinear Ordinary Differential Equations, Oxford University Press, 2007.
[20] P. Cvitanović, R. Artuso, R. Mainieri, G. Tanner, G. Vattay, Chaos: Classical and Quantum, I: Deterministic Chaos, 2009, ChaosBook.org.
[21] K.J. Badcock, M.A. Woodgate, B.E. Richards, Hopf bifurcation calculations for a symmetric airfoil in transonic flow, AIAA J. 42 (5) (2004) 883–892.

[22] K. Ramesh, J. Murua, A. Gopalarathnam, Limit-cycle oscillations in unsteady flows dominated by intermittent leading-edge vortex shedding, J. Fluids Struct. 55 (2015) 84–105.
[23] B.H.K. Lee, S.J. Price, Y.S. Wong, Nonlinear aeroelastic analysis of airfoils: bifurcation and chaos, Prog. Aerosp. Sci. 35 (1999) 205–334.
[24] E.H. Dowell, R. Clark, D. Cox, H.C. Curtiss Jr., J.W. Edwards, K.C. Hall, D.A. Peters, R. Scanlan, E. Simiu, F. Sisto, T.W. Strganac, A Modern Course in Aeroelasticity, 4th edition, Kluwer Academic Publishers, Berlin, Germany, 2004 (Chapter 11).
[25] O. Bendiksen, Influence of shocks on transonic flutter of flexible wings, in: 50th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, American Institute of Aeronautics and Astronautics, 2009.
[26] M. Manetti, G. Quaranta, P. Mantegazza, Numerical evaluation of limit cycles of aeroelastic systems, J. Aircr. 46 (2009) 1759–1769.
[27] S. Fichera, S. Ricci, Freeplay-induced limit-cycle oscillations in a T-tail: numerical vs experimental validation, J. Aircr. 52 (2) (2014) 486–495.
[28] S. Haykin, Neural Networks and Learning Machines, Pearson International Edition, 2009.
[29] G. Romanelli, E. Serioli, P. Mantegazza, A “free” approach to computational aeroelasticity, in: 48th AIAA Aerospace Sciences Meeting and Exhibit, 2010.
[30] J. Blazek, Computational Fluid Dynamics: Principles and Applications, Elsevier, 2001.
[31] S. Haykin, Neural Networks and Learning Machines: A Comprehensive Foundation, Prentice Hall, 2008.
[32] M. Mattaboni, G. Quaranta, P. Mantegazza, Active flutter suppression for a three-surface transport aircraft by recurrent neural networks, J. Guid. Control Dyn. 32 (2009) 1295–1307.
[33] Y. Bengio, P. Simard, P. Frasconi, Learning long-term dependencies with gradient descent is difficult, IEEE Trans. Neural Netw. 5 (1994) 157–166.
[34] D. Marquardt, An algorithm for least-squares estimation of nonlinear parameters, SIAM J. Appl. Math. 2 (1963) 431–441.
[35] D. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Kluwer Academic Publishers, 1989.
[36] S. He, Z. Yang, Y. Gu, Transonic limit cycle oscillation analysis using aerodynamic describing functions and superposition principle, AIAA J. 52 (7) (2014) 1393–1404.
[37] R.L. Bisplinghoff, H. Ashley, R.L. Halfman, Aeroelasticity, Dover, 1955.
[38] P. Masarati, M. Morandini, P. Mantegazza, An efficient formulation for general-purpose multibody/multiphysics analysis, J. Comput. Nonlinear Dyn. 9 (4) (2014).