Journal of Fluids and Structures 131 (2024) 104202

Contents lists available at ScienceDirect

Journal of Fluids and Structures

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

Recurrent graph convolutional multi-mesh autoencoder for unsteady transonic aerodynamics

David Massegur [*,1], Andrea Da Ronch[2]

Faculty of Engineering and Physical Sciences, University of Southampton, Southampton, SO16 7QF, United Kingdom

ARTICLE INFO

Keywords:
Geometric deep learning
Recurrent neural network
Autoencoder
Computational fluid dynamics
Unsteady aerodynamics
Graph convolutional network
Multi mesh
BSCW wing

ABSTRACT

Unsteady, high-fidelity aerodynamic load predictions around a three-dimensional configuration will remain computationally expensive for the foreseeable future. Data-driven algorithms based on deep-learning are an attractive option for reduced order modelling of complex, nonlinear systems. However, a dedicated approach is needed for applicability to large and unstructured domains that are typical in engineering. This work presents a geometric-deep-learning multi-mesh autoencoder framework to predict the spatial and temporal evolution of aerodynamic loads for a finite-span wing undergoing different types of motion. The novel framework leverages on: (a) graph neural networks for aerodynamic surface grids embedded with a multi-resolution algorithm for dimensionality reduction; and (b) a recurrent scheme for time-marching the aerodynamic loads. The test case is for the BSCW wing in transonic flow undergoing a combination of forced-motions in pitch and plunge. A comprehensive comparison between a quasi-steady and a recurrent approach is provided. The model training requires four unsteady, high-fidelity aerodynamic analyses which require each about two days of HPC computing time. For any common engineering task that involves more than four cases, a clear benefit in computing costs is achieved using the proposed framework as an alternative predictive tool: new cases are computed in seconds on a standard GPU.

1. Introduction

The high resources required in high-fidelity computational fluid-dynamics (CFD) solutions are in contrast with the ever more stringent timescales of an engineering design cycle, as common among other examples in the aerospace and motorsport environments. The disproportion between the required computing costs and the desired fast turnaround times of product development is exacerbated when unsteady problems are considered, such as in buffet envelope search or gust response (Righi et al., 2021; Raveh, 2007).

Recent advances in machine learning (ML) algorithms result convenient as reduced order models (ROM) to accelerate simulation turnarounds, given their ability to discover highly complex and nonlinear systems from data. In a previous contribution by Massegur et al. (2022), a ML-based model to predict the aerodynamic loads of a wing section was coupled to a general-purpose structural solver to predict the transonic flutter boundary.

Building upon our prior research involving a two-dimensional (2D) setup, we are herein interested in providing solutions to the modelling challenges involved with unsteady and three-dimensional (3D) configurations, more applicable to real-world purposes but

* Corresponding author.
E-mail address: David.Massegur-Sampietro@southampton.ac.uk (D. Massegur).
[1] PhD Student.
[2] Professor.

https://doi.org/10.1016/j.jfluidstructs.2024.104202
Received 4 March 2024; Received in revised form 3 July 2024; Accepted 8 October 2024
Available online 28 October 2024

with added complexity. Nonlinear flow features and interactions in 3D systems are more abundant and richer in spatio-temporal scales than in 2D (Anderson, 2016; Pope, 2000). Therefore, ROM approaches that work well in 2D tasks (Massegur et al., 2022) may be inappropriate to solve 3D problems. As a result, to address spatio-temporal aerodynamic predictions on 3D wing surfaces, we resort to deep learning (Goodfellow et al., 2016), the branch of ML that investigates neural networks (NN), as these are particularly suited for inference of highly nonlinear systems.

Another central problem with reduced-order modelling for aerodynamic analyses entails dealing with large and irregular domains resulting from the discretisation employed in CFD grids (Brunton and Kutz, 2019). In this case, common nonlinear ROM techniques, including Volterra (Silva, 1993), Kriging (Glaz et al., 2010) or dense neural networks (DNN) (Mannarino and Mantegazza, 2014; Wang et al., 2020) result ill-suited. To solve the issue of large spatial domains, a dimensionality reduction technique is sought. Neural-network based dimensionality reduction techniques, known as autoencoders (AE), result more attractive than the classical proper orthogonal decomposition (POD) (Park et al., 2013; Ribau et al., 2021) because of their ability to execute nonlinear projections onto the compressed space.

Related to unstructured domains, renowned convolutional neural networks (CNN) (Sureshbabu et al., 2023; Morimoto et al., 2021), regardless if embedded in an autoencoder approach (CNN-AE) (Rozov and Breitsamter, 2021), are problematic because they require Euclidean domains (cartesian grids) that are generally incompatible with the unstructured meshes typical in CFD. Instead, we resort to geometric deep learning, the umbrella of graph neural networks (GNN) which involve convolutional methods adapted to non-Euclidean spaces (Bronstein et al., 2021, 2017). GNNs for steady-state predictions were proposed by Hines and Bekemeyer (2023), Ogoke et al. (2021). Regarding unsteady investigations, while Pfaff et al. (2020) did not address the dimensionality issue, Han et al. (2022) adopted a simplistic autoencoder scheme for 2D cases. To the best of our knowledge, spatio-temporal predictions for dynamically moving aircraft components, using large and unstructured meshes, has not been addressed.

The present work builds upon a previous contribution by Massegur and Da Ronch (2024a), where a steady-state graph convolutional network embedded in a multi-mesh autoencoder (GCN-MM-AE) was implemented. The scheme is particularly suited to CFD meshes and is reminiscent of the multi-grid schemes for solution of partial differential equations (McCormick, 1987; Smith, 1990).

Since we are herein interested in dynamic simulations, embedding the GCN-MM-AE model in a time-dependent framework is sought. To account for temporal predictions, recurrent neural networks (Graves, 2013) can be thought within a time-marching approach. However, a critical issue with time-dependent methods is the error accumulation in long-term forecast (Font et al., 2021; Beck and Kurz, 2020). A simplistic forward Euler scheme with input perturbation from random noise was proposed by Pfaff et al. (2020), whereas a transformer model was adopted by Han et al. (2022). The transformer algorithm (Vaswani et al., 2017) is renowned for faithful long-term forecasting but it results computationally expensive. Therefore, the challenges involved in temporal predictions present several open questions and we addressed these devising our own recurrent scheme.

We present a recurrent graph-convolutional multi-mesh autoencoder for unsteady aerodynamics prediction on aircraft surfaces. The unique contribution of this work is applicability to non-Euclidean domains, a multi-resolution embedding for model efficiency, a time-marching scheme with long-term error considerations and a building-block implementation to easily adapt the framework to different tasks: coarse-data reconstructions, quasi-steady or unsteady regimes. For the validation of the developed methodology we used the benchmark supercritical wing (BSCW) from the Aeroelastic Prediction Workshop (AePW) (Heeg et al., 2015, 2016; Heeg and Chwalowski, 2017) at transonic regime as test case. The wing is subject to forced motion of two degrees of freedom. As a result, the model predicts the unsteady pressure field on the surface of the moving wing.

The paper continues with a description of the test case and the complex physics to deal with in Section 2. Section 3 reports the methodology developed in the present study. Results found with the different modelling approaches are reported in Section 4 followed by the concluding remarks of Section 5.

2. Test case: Benchmark supercritical wing

The BSCW test case has been investigated for aeroelastic studies across the various AePW sessions (Heeg et al., 2015). The NASA SC(2)0414 aerofoil is used as wing section. The wing is rigid with two degrees of freedom in pitch α and plunge $\xi = h/b$, as illustrated in Fig. 1 Panel (a). The reference geometric chord is $c = 0.4064$ m, with b indicating the semichord, and the surface area $S = 0.3303$ m^2. The origin for moment calculations is at mid chord and the non-dimensional time is defined as $\tau = t\,U_\infty/b$, with U_∞ the freestream speed. The cross sections in Fig. 1 will be used for analysis of the results in Section 4.

The open-source SU2 7 solver was used for the reference aerodynamic analyses. We addressed the analyses on the coarse tetrahedral mesh from the AePW.[3] Details of the mesh are illustrated in the other three Panels of Fig. 1. The mesh consists of 2.97 million grid nodes, of which 86k on the wing surface, shown in red. The mesh choice was based on a compromise between adequate spatial resolution and computational efficiency. The one-equation Spalart–Allmaras (SA) turbulence model was used. To achieve flow-solution convergence, we adopted the JST convective numerical method, the biconjugate gradient stabilisation as linear solver, Green-Gauss method for the spatial gradients, Euler implicit as time discretisation with a time step of $dt = 2 \cdot 10^{-4}$ s, Ilu preconditioner and $CFL = 5$. Freestream flow conditions were set at three different Mach numbers $M_\infty = [0.74, 0.80, 0.84]$, with fixed $Re = 4.49 \cdot 10^6$, $q_\infty = 8000$ Pa and $\alpha_\infty = 0$ deg. The flow is first solved around the static wing and then the grid-movement solver built in SU2 is enabled to address the rigid body motions.

[3] https://nescacademy.nasa.gov/workshops/AePW2/public/BSCW/analystsInfo

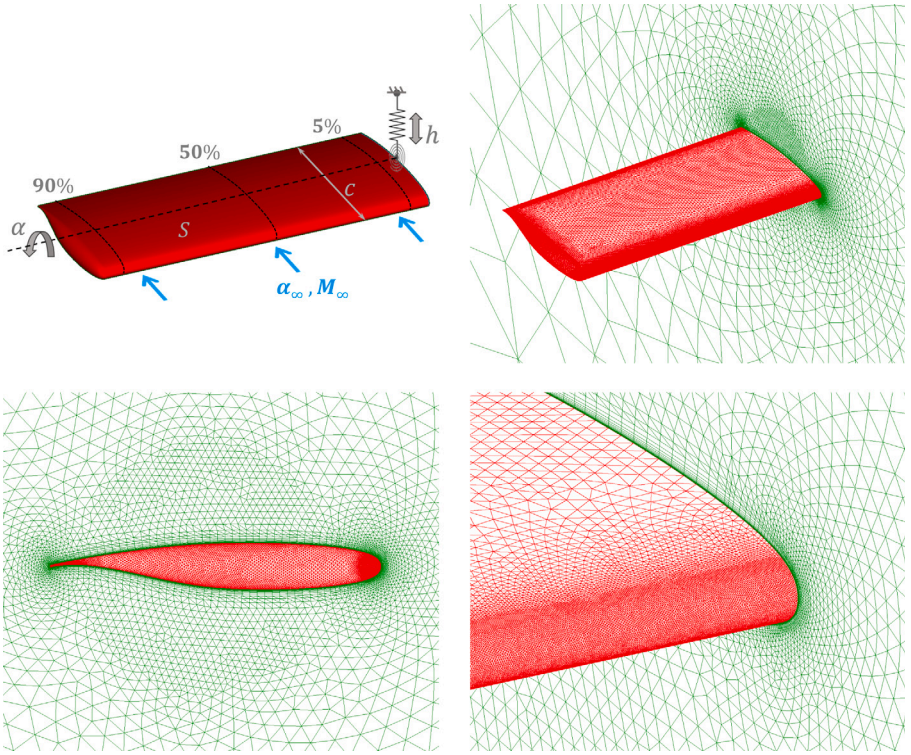


Fig. 1. Details of the geometry and the unstructured tetrahedral mesh for the BSCW model, elastically restrained in pitch and plunge. Symmetry-plane (green) and wing (red) boundaries are shown. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

2.1. Forced-motion simulations

Forced-motion CFD responses were used to generate our ROMs. To achieve an efficient implementation, we minimised the number of necessary training signals. To this aim, proper consideration on the type of motion signal to adopt is sought. The aerofoil flutter boundary study from Massegur et al. (2022) indicated that using free aeroelastic responses to some initial conditions as training signals resulted adequate but was limited to a single structural configuration. By contrast, the widely used Schroeder-phased multi-harmonic signals (Wojtczak and Oxenham, 2009; Lentz and Leek, 2001; Sahinkaya et al., 2002) failed to capture the structural damping correctly. Herein, we are interested in a single aerodynamic ROM applicable to various motions than those used to generate the model, e.g. for different reduced frequencies due to structural modifications. As a result, we propose a modification of the common Schroeder signal by adding a linear decay of the amplitude over time, resembling simplified structural damping. This adaptation enriches the dataset as it includes varying amplitudes for each Schroeder harmonic over time, therefore exciting a wider range of dynamics, whereas with the original Schroeder formulation, the signal is simply periodic, i.e. repetitive.

The proposed modified Schroeder formulation is defined as:

$$\hat{a}(t, a, \omega) = \sum_{i=1}^{n_m} a(t) \sin(i\,\omega\, t + \phi_i) \tag{1}$$

where $\hat{a}(t)$ is the total instantaneous amplitude of the input signal, in our case pitch or plunge motions, n_m the total number of harmonics (or modes), and ω the first-harmonic frequency and $a(t)$ its amplitude. In the original Schroeder formulation, $a(t)$ is a constant, while in our modified version, the harmonic amplitudes decay linearly over time:

$$a(t) = a(t_0)\,\frac{a(t_{\text{end}}) - a(t_0)}{t_{\text{end}} - t_0}\,(t - t_0) \tag{2}$$

with t_0 and t_{end}, respectively, the initial and final time instants. The initial signal amplitude $a(t_0)$ were prescribed for each signal and the total simulation time was set to $t_{\text{end}} = 2$ s. The final amplitude was chosen as $a(t_{\text{end}}) = 0.1\ a(t_0)$ to ensure significant decay of the amplitudes by the end of the response.

Using the proposed modified Schroeder formulation in Eq. (1), we specify different signals for combined pitch and plunge modes:

$$\begin{cases} \alpha(t) = \hat{a}_\alpha(t, a_\alpha, \omega_\alpha) \\ \xi(t) = \hat{a}_\xi(t, a_\xi, \omega_\xi) \end{cases} \tag{3}$$

Table 1
Phase for each harmonic mode of the Schroeder signal.

Mode i	1	2	3	4	5	6	7	8	9
ϕ_i [deg]	0	−0.6981	−2.0944	−4.1888	−6.9813	−10.4720	−14.6608	−19.5477	−25.1328

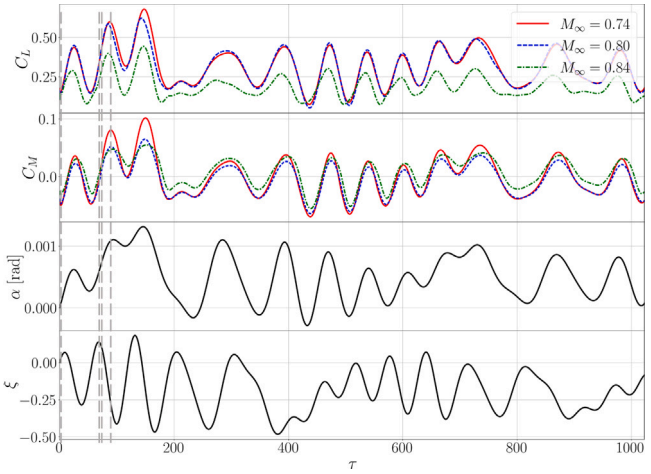


Fig. 2. Aerodynamic response of the BSCW wing using modified Schroeder-phased signals for pitch and plunge motions, at the three different Mach numbers, $Re = 4.49 \cdot 10^6$, $q_\infty = 8000$ Pa and static $\alpha_\infty = 0$ deg. $M_\infty = 0.74$ in dash style, $M_\infty = 0.80$ in dot style, $M_\infty = 0.84$ in dash-dot style.

Therefore, the first-harmonic frequencies ω_α and ω_ξ were assigned for each motion mode and the first-harmonic amplitudes, $a_\alpha(t)$ and $a_\xi(t)$, are defined by Eq. (2) given the user-defined initial amplitudes:

$$\begin{cases} a_\alpha(t) = a(t, a_\alpha(t_0)) \\ a_\xi(t) = a(t, a_\xi(t_0)) \end{cases} \tag{4}$$

We adopted $n_m = 9$ harmonics to define the response signals as it covered sufficient range of representative frequencies. The reason for choosing combined frequencies is for the ROM to be able to learn the response from multiple frequency excitation with a single training signal (Da Ronch et al., 2013). Table 1 reports the phase values for each mode. Refer to Table 8 in Appendix A.1 for the definition of the various signals used to generate and validate the ROMs.

Fig. 2 showcases an example of CFD aerodynamic response at the three Mach numbers and given pitch-and-plunge motions based on modified Schroeder-phased signals. The first-harmonic of this training signal has frequency and initial amplitude set to $\omega_\alpha = 1.4$ Hz, $a_\alpha(t_0) = 0.8$ deg and $\omega_\xi = -1.6$ Hz, $a_\xi(t_0) = -0.098$ for pitch and plunge, respectively. The instantaneous lift and pitching moment coefficients, C_L and C_{M_y}, are reported for each Mach number to assess the variation of the aerodynamic response caused by the dynamic input motion and across the various freestream regimes. We observe that the lift coefficient correlates with the combination of pitch and plunge modes. However, a decay of the C_L at the highest Mach number is visible, due to wing stall caused by the shock-induced boundary layer separation. The C_{M_y} also correlates with the motion inputs. At $M_\infty = 0.84$, the loss of lift is compensated by the centre of pressure shifting towards the leading edge, resulting in pitching moment magnitudes similar to the other Mach conditions.

We are interested in predicting the time evolution of the pressure coefficient distribution. We extracted snapshots of the flow field corresponding to the four time steps indicated by vertical lines in Fig. 2. The selected C_p fields are shown in Fig. 3 with the Mach number ($M_\infty = 0.74$, 0.80 and 0.84) column-wise and the time instants span-wise. This comparison evidences the impact of the Mach number on the location and strength of the shock wave. At $M_\infty = 0.74$, far left column, we observe an elliptic shock-wave structure along the span, due to low wing aspect ratio. The time evolution presents large fluctuations of the shock wave location in chord. On the contrary, at higher Mach numbers the shock wave remains on average at a fixed position. At $M_\infty = 0.80$, middle column, the shock wave line is trapezoidal and there is intermittent boundary-layer separation downstream and subsequent re-attachment, as denoted by varying C_p values downstream of the shock wave. At $M_\infty = 0.84$, far right column, the shock-induced separation is permanent, evidenced by a flat C_p distribution aft the shock wave.

The observed nonlinear and localised phenomena reveal the complex physics involved with this use case that would not be present on a 2D problem. From a computing standpoint, one unsteady CFD simulation requires over 10,000 CPU hours on a high-performance cluster. These considerations motivate leveraging deep-learning for our spatio-temporal prediction model.

3. Methodology

In this section we describe the developed methodology for spatio-temporal aerodynamic predictions. The idea is the generation of a neural-network function f_{NN} which maps specific user-defined inputs s, i.e. the motion signals in the BSCW case, to desired

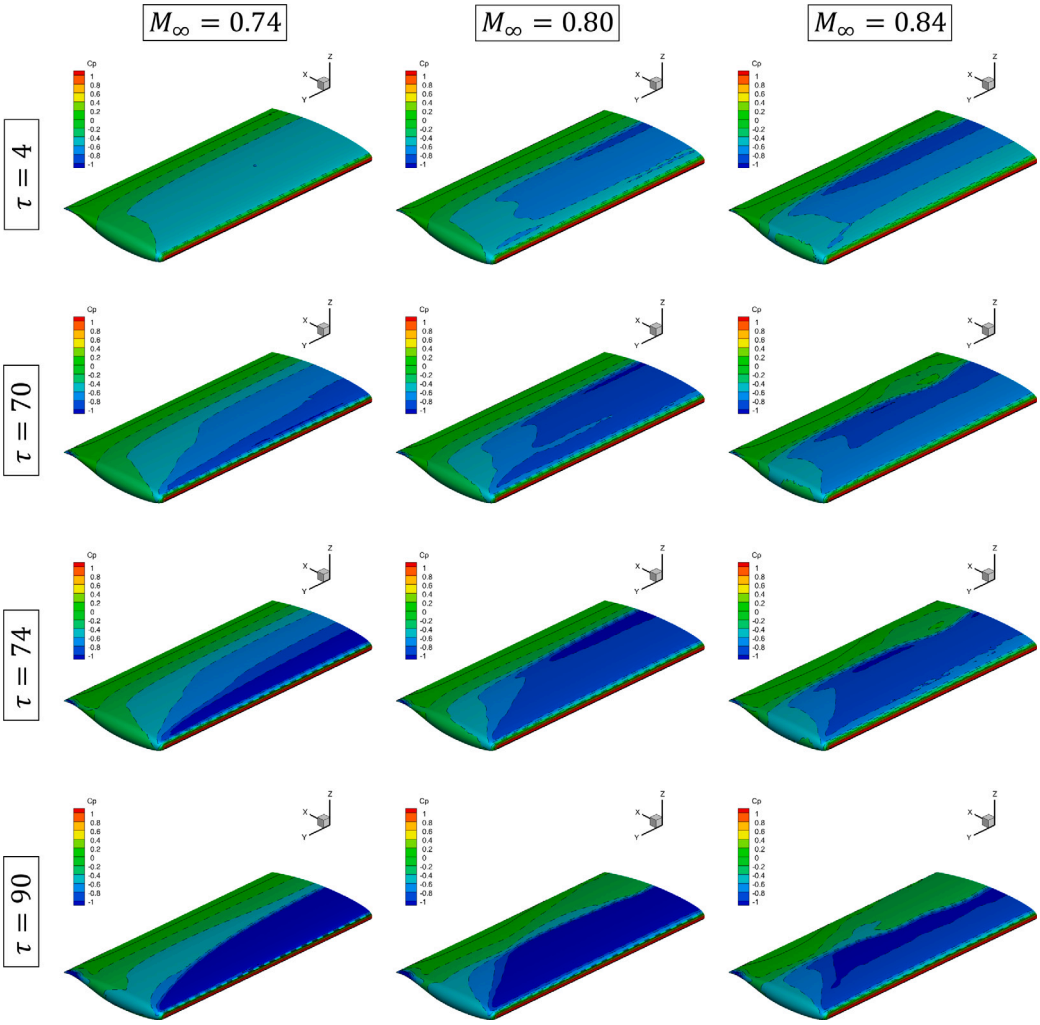


Fig. 3. Snapshots of the pressure coefficient at three different Mach numbers (columns) and time instants (rows), selected from the responses in Fig. 2. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

target $\boldsymbol{Y}_i$ fields on the surface mesh $\mathcal{S}$, such as the pressure distribution. We define the ROM to output the solution with respect to the initial static condition:

$$\boldsymbol{Y}_i(\tau) = \boldsymbol{Y}_i(\tau_0) + f_{\text{NN}}\left(\boldsymbol{s}(\tau), \boldsymbol{x}_i, \boldsymbol{\Theta}\right) \quad \forall\, i \in \mathcal{S} \tag{5}$$

where the $[x, y, z]$ coordinates $\boldsymbol{x}_i \in \mathbb{R}^{n\times 3}$ of the mesh nodes $i \in \mathcal{S}$, with n_n the input mesh node count, are also embedded to enhance the spatial mapping. $\boldsymbol{\Theta}$ represents the set of trainable model parameters within the various layers. From here onward, we will omit the static contribution $\boldsymbol{Y}_i(\tau_0)$ in the expressions but it is understood that the developed aerodynamic models are always relative to this condition.

Since we are dealing with wings subject to rigid body motion, the surface mesh remains unchanged. It implies that we can operate on the same mesh throughout the whole time history. Therefore, as introduced in Section 1, it is possible to build this work upon a previous contribution (Massegur and Da Ronch, 2024a) for the spatial modelling. That work presented a multi-mesh graph convolutional autoencoder framework, which we named GCN-MM-AE, for the steady-state prediction of the pressure and shear-stress distributions on an aircraft surface. The following section provides a brief description of the developed methodology. Refer to the original paper for the complete formulation.

3.1. Graph convolutional multi-mesh autoencoder

The implemented method leverages geometric deep learning (Bronstein et al., 2021), i.e. GNNs, and dimensionality reduction on the surface of the CFD mesh, which can be represented as a graph. Target spatial quantities $\boldsymbol{y}_i$ and position coordinates $\boldsymbol{x}_i$ are

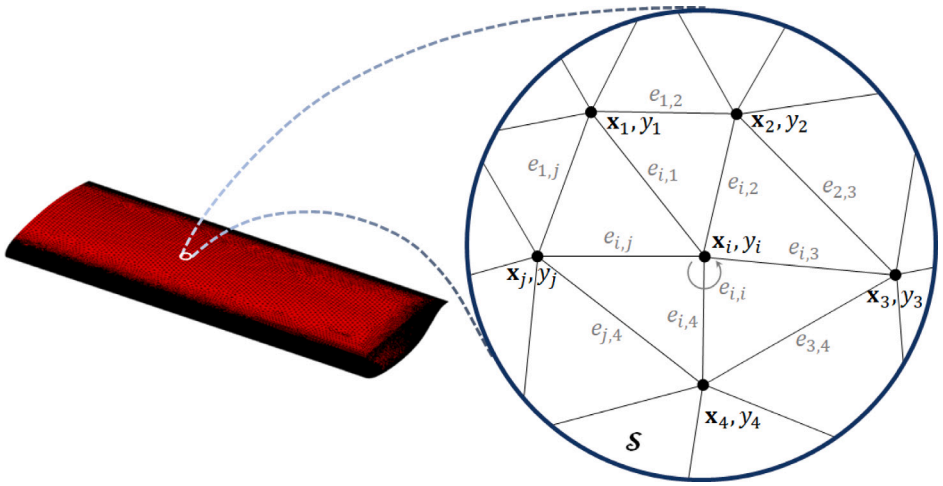


Fig. 4. BSCW mesh represented as a graph with node features and edge weights.

assigned on the graph nodes ; and user-defined weights e_{ij}, on the edges of connected nodes. Fig. 4 illustrates a generic graph representation, including node-based features, coordinates and edge weights.

From the family of GNNs, we chose the widely used graph convolutional network (GCN) layer (Kipf and Welling, 2017), defined as:

$$\text{GCN:} \quad g(\boldsymbol{y}) = h\left(\boldsymbol{\theta}^T \, \hat{\boldsymbol{D}}^{-\frac{1}{2}} \hat{\boldsymbol{A}} \, \hat{\boldsymbol{D}}^{-\frac{1}{2}} \, \boldsymbol{y} + b\right) \tag{6}$$

with $\boldsymbol{\theta}$ a layer-specific trainable weight vector, b a constant weight term and $\boldsymbol{y}$ the node-based input vector at each node of the mesh $\mathcal{S}$. $\hat{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{I}$ corresponds to the adjacency matrix, with $\boldsymbol{A} = e_{ij}$, and $\hat{\boldsymbol{D}} = \text{diag}\left(\sum_{j \neq i} e_{ij} + 1\right)$, the diagonal degree matrix, this is the sum of the edge weights connected to target node i. The connectivity matrix is largely sparse. Consequently, its arrangement in sparse form is convenient: for each edge, the indices of the connected node pairs i, j and the respective weight value e_{ij} is included, resulting in a more compact size $\hat{\boldsymbol{A}} \in \mathbb{R}^{n_{e_n} \times 3}$. Furthermore, we chose as nonlinear activation $h(x)$ the PReLU function (He et al., 2015):

$$\text{PReLU:} \quad h(y) = \begin{cases} y & \text{if} \quad y \geq 0 \\ \beta y & \text{if} \quad y < 0 \end{cases} \tag{7}$$

where β is a learnable parameter independent for each channel of the input vector.

With the GCN operator, each node i can receive influence from the connected nodes only, resulting useful to extract localised features. By contrast, global flow features can be captured with GCNs via propagation of information across the mesh. To achieve this, an impractically large network of GCN layers would be required. To circumvent the burden of a big network, especially in the case of large meshes, an autoencoder approach for dimensionality reduction (Liu et al., 2023; Hinton and Salakhutdinov, 2006) was developed. In brief, the GCN layers are embedded in a novel multi-mesh (MM) scheme, resembling the 2-level multi-grid V-cycle method to solve partial differential equations (Smith, 1990; McCormick, 1987). A schematics of the implemented multi-mesh cycle is illustrated in Fig. 5, with the original mesh represented as $\mathcal{S}_\text{n}$ and the coarsened mesh, as $\mathcal{S}_\text{r}$. The MM autoencoder coarsens the mesh while extracting crucial features with the GCNs, process known as encoder. In the decoder process, the compressed latent states are embedded back onto the original mesh.

Table 2 reports the mesh sizes (number of nodes) adopted in the multi-resolution scheme for the BSCW test case. The nodes carried over to the coarse mesh were selected via a probability function based on the mesh density distribution. The selection probability distribution was assigned to reasonably keep the original mesh topology but without losing excessive resolution in the originally coarse regions. With this procedure, a mesh compression ratio of 16 was achieved. To proceed with GCN operations on the coarse mesh, the graph connectivity is reconstructed by executing a search loop of nearby remaining nodes around each unremoved node. Last, to operate between mesh levels, the information must adequately be interpolated from one mesh to the other. The interpolation consists of a matrix $\boldsymbol{\mathcal{I}}_{\mathcal{S}_\text{n} \to \mathcal{S}_\text{r}} : \mathbb{R}^{n_n} \to \mathbb{R}^{n_r}$ to map a spatial field $\boldsymbol{y}$ from the source grid $\mathcal{S}_\text{n}$ to the target grid $\mathcal{S}_\text{r}$, with both grids discretising the same spatial domain:

$$\text{Interpolation matrix:} \quad \boldsymbol{y}_j = \boldsymbol{\mathcal{I}}_{\mathcal{S}_\text{n} \to \mathcal{S}_\text{r}} \, \boldsymbol{y}_i \quad \forall j \in \mathcal{S}_\text{r}, \forall i \in \mathcal{S}_\text{n} \tag{8}$$

The interpolation matrix is not invertible. Consequently, two independent interpolation matrices are required for the reduction and recovery operations, respectively. We implemented a polynomial-based weighted moving least squares scheme to construct each matrix (Quaranta et al., 2005; Joldes et al., 2015). The adoption of the MM scheme is advantageous for:

1. Limiting the model memory burden by operating in a compressed space.

Table 2
Surface mesh size (number of nodes and edges) of the two multi-mesh levels for the BSCW test case.

Original mesh		Coarsened mesh		Compression
nodes n_n	edges n_{e_n}	nodes n_r	edges n_{e_r}	ratio $\frac{n_n}{n_r}$
86,840	520,665	5427	26,238	16.0

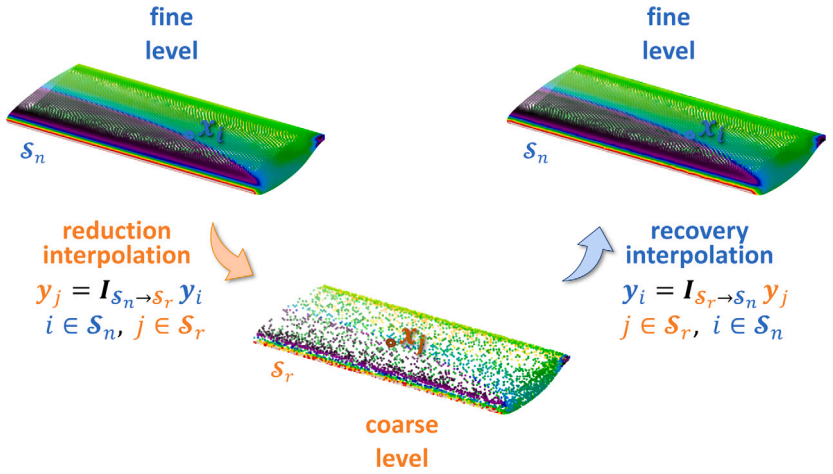


Fig. 5. Schematics of the 2-level multi-mesh cycle for the BSCW wing, involving one coarsening level and interpolation between meshes.

2. Enabling the GCN blocks to extract either local and global features, in analogy with the pooling operations in CNNs (Morimoto et al., 2021).
3. Efficiently propagate influence between far-away nodes, avoiding the need for a deep network and its potential numerical instabilities.

Note that the introduced GCN-MM-AE model solves the spatial modelling only. Since we are interested in dynamic responses, adaption of this method in a time-dependent framework is sought. Different approaches can be thought to account for the temporal modelling. Here we developed two distinct techniques: a quasi-steady and a recurrent scheme. We lay down a comprehensive comparison analysis between the two and report the benefits and disadvantages for each approach.

3.2. Time-dependent prediction framework

First, we formalise the methodology for the recurrent model, involving a time-marching process. Subsequently, we derive the quasi-steady implementation from a simplification of the recurrent scheme.

The idea for a time-marching process is to let the past flow condition influence the next-step solution. This is achieved by defining the next-step prediction $\boldsymbol{Y}_i(\tau)$ as a function of the external inputs as well as the solution from the previous time step $\tau - \mathrm{d}\tau$:

$$\text{Recurrent functional:} \quad \boldsymbol{Y}_i(\tau) = f_{\mathrm{NN}}\left(\boldsymbol{s}(\tau), \boldsymbol{x}_i, \boldsymbol{Y}_i(\tau - \mathrm{d}\tau), \boldsymbol{\Theta}\right) \tag{9}$$

In particular, the target prediction for the BSCW consists of the pressure coefficient field $\boldsymbol{Y}_i = [C_{p_i}] \in \mathbb{R}^{n \times 1}$ as a function of the pitch and plunge motion inputs. Alternatively, one could be interested to predict the shear stresses or the wall temperature. For model compactness, it is convenient to adopt the modal coordinates and time derivatives $\boldsymbol{s} = [\alpha, \dot{\alpha}, \ddot{\alpha}, \dot{\xi}, \ddot{\xi}]$ as input variables as opposed to the grid-point displacements. As a result, since we are dealing with rigid-body motions, the graph can be kept fixed in its static position, significantly reducing the computational complexity of having to update the graph coordinates and connectivity at each time instant. Furthermore, the plunge state ξ can be excluded since the instantaneous vertical position does not exert any aerodynamic effect. Note that, although the modal coordinates are global quantities, these are recast as uniform fields into the nodes of the graph. As a result, the dimensionality of the motion inputs becomes $\boldsymbol{s} \in \mathbb{R}^5 \to \mathbb{R}^{n \times 5}$.

The recurrent algorithm is sought to generate an embedding between the current motion inputs and the previous solutions to execute the next-step prediction (Han et al., 2022). Deep learning provides a suite of recurrent neural networks (RNN) (Goodfellow et al., 2016; Brunton et al., 2020) particularly designed for time-series forecasts. The idea here is to embed a RNN based algorithm with the GCN-MM-AE model. With the choice of modal states, the question is:

- What is the most adequate combination between nodal-based (the previous-step pressure field) and global-based (wing motion states) as input variables?

Direct concatenation of the two input classes as proposed by Rozov and Breitsamter (2021) can be problematic for potential numerical divergence (Beck and Kurz, 2020). To mitigate the issue of long-term error accumulation, we adopted the gated recurrent unit (GRU) (Cho et al., 2014) for the recurrent embedding. Formulation of the GRU operator is provided in Appendix A.2. The GRU provides capability of carrying over long-term memory on subsequent predictions as other more common units, i.e. the long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997) and the Transformer (Vaswani et al., 2017), but with the advantage of being less memory burdensome. This feature makes the GRU more advantageous to limit the computing burden (Chung et al., 2014). To limit model memory requirements, a plausible location for embedding the modal inputs is aft the MM reduction step, wherein the spatial size is the smallest.

The devised model architecture, featuring separate input branches, the MM autoencoder and the recurrent blocks, results rather complex for the training. To alleviate the model generation process and to avoid potential instabilities or memory errors, a novel two-module training sequence was adopted, building on the approach from Han et al. (2022). The architecture of the designed recurrent GCN-MM-AE for spatio-temporal predictions is depicted in Fig. 6. The different modules represented by each Panel are trained sequentially. The autoencoder in isolation is generated first, Panel (a). This model addresses compression and subsequent reconstruction of the C_p snapshots, this is:

$$\text{Autoencoder functional:} \quad \boldsymbol{Y}_i(\tau) = f_{\text{NN}}\left(\boldsymbol{Y}_i(\tau), \boldsymbol{x}_i, \boldsymbol{\Theta}\right) \tag{10}$$

The autoencoder is adequate to encode the pressure field in the latent state. To perform new predictions, we subsequently leverage the same autoencoder, by means of transfer learning, in a second module which also incorporates the motion inputs to execute the time-marching stepping. This second model is illustrated in Panel (b) of Fig. 6. The C_p field from the previous time step $\tau - \mathrm{d}\tau$ is input to the encoder, previously trained and frozen. A second branch encodes the modal states $\boldsymbol{s}(\tau)$. The two processed latent states are input to the GRU core to execute the new-state update. This new state is subsequently recovered by the decoder, also pre-trained in the first module, to output the C_p field at the current step τ.

With this strategy, the second module involves only the training of the second encoder, related to the motion inputs, and the recurrent unit. The choice of this two-module procedure is motivated by:

1. Reducing the computing burden in the training process by means of a sequential model optimisation.
2. A pre-trained autoencoder together with the building-block implementation facilitates verification of different external input types and recurrent units, including adaptation to a quasi-static scheme.

3.2.1. Multi-step training sequence

As introduced in Section 1, a critical problem with time-marching simulations is the long-term error accumulation. Even if prediction accuracy is high in a single time step, the error accumulation over successive iterations can lead to drastic solution divergence (Thuerey et al., 2021). This phenomenon will unlikely be correctly addressed if the training loss execution involves single time-marching step sequences. Therefore, to mitigate long-term error accumulation, a multi-step sequence was adopted for the optimisation of the model parameters: the dataset is arranged in time sequences of a prescribed number of steps k. The model is sequentially called to recurrently process the entire mini sequences, letting the prediction errors accumulate. Once each k-step sequence is completed, the total loss is computed to execute the backpropagation and model parameter update steps. Fig. 7 illustrates a diagram of this multi-step training sequence. In particular, we chose a sequence length of $k = 3$ steps because it was found a good compromise between training burden and adequate prediction results throughout the time history.

In addition, we also adopted a solution recommended by Pfaff et al. (2020) which consists of perturbing each input signal with a normally distributed random noise of standard deviation 10% the maximum magnitude of the signal.

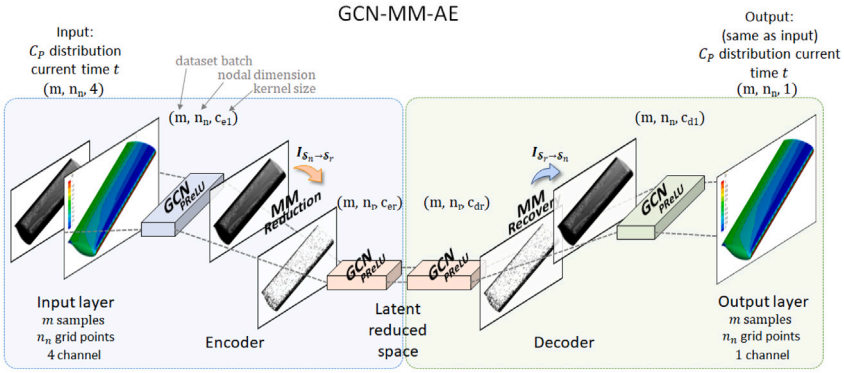
3.3. Quasi-steady prediction framework

In a quasi-steady scheme the aerodynamic response is uniquely dependent on the wing motion, excluding the influence from the previous flow condition in Eq. (9):

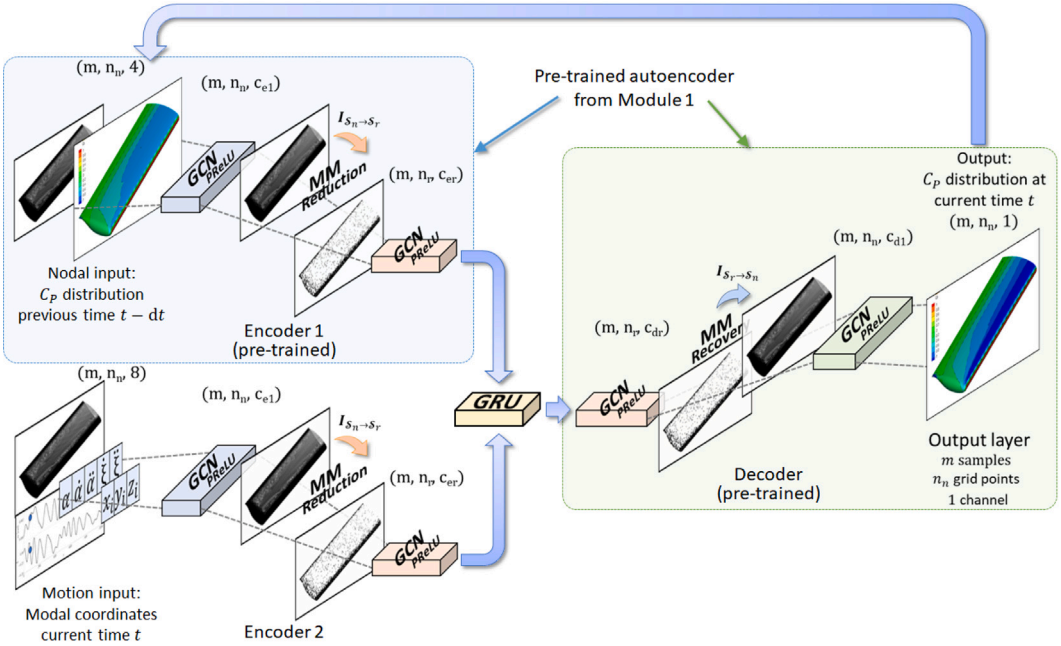
$$\text{Quasi-steady functional:} \quad \boldsymbol{Y}_i(\tau) = f_{\text{NN}}\left(\boldsymbol{s}(\tau), \boldsymbol{x}_i, \boldsymbol{\Theta}\right) \tag{11}$$

A quasi-steady approach assumes instantaneous reaction to body motion as it does not leverage the time history of the flow. Consequently, response variability caused by unsteady effects cannot be captured. On the other hand, by disregarding the past flow condition at each time-step prediction, the risk of long-term error accumulation is reduced.

The quasi-steady scheme can be easily derived with a modification of the recurrent framework just described. The encoder used to embed the previous-step solution and the GRU core in the second module of the recurrent framework, Fig. 6 Panel (b), are disregarded. As a result, the quasi-steady model architecture is formed by the second, input-signal encoder followed by the pre-trained and frozen decoder. This approach is reminiscent of the POD reconstruction and the Galerkin method for interpolation of the POD coefficients (Luchtenburg et al., 2009). For clarity, Table 3 summarises the input and output quantities adopted for each modelling approach. The dimensions of the tensors are also included. Note how each model considers as additional inputs the coordinates of the fixed mesh $\boldsymbol{x}$, aimed at inferring the spatial mapping, and the connectivity $\hat{\boldsymbol{A}}$, required by the GCN layers to execute the message passing through the edges. In this particular case, the global inputs $\boldsymbol{s}$ involve the pitch and plunge motion states. However, with our implementation, the framework is able to accommodate other types of inputs, such as flow conditions (Mach



(a) Module 1: autoencoder for the C_p compression and reconstruction.



(b) Module 2: next-step predictor for the time-marching process. The autoencoder is pre-trained in Module 1.

Fig. 6. Schematic of the two-phase recurrent GCN-MM-AE framework used for time-dependent responses.

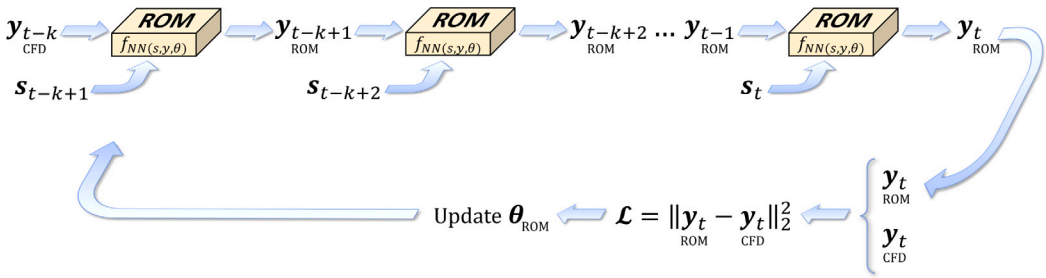


Fig. 7. Schematic of the multi-step sequence for training the recurrent model.

number, Reynolds number, turbulence intensity or freestream temperature). Alternative inputs could involve design parameters in the context of a parametric design optimisation. Hence, within the same framework, different aerodynamic scenarios are addressed.

Table 3
Relation of inputs and outputs involved in each model approach, including the respective dimensionality. The prediction quantity involves the pressure coefficient $\boldsymbol{Y} = [C_p]$, global inputs are the pitch and plunge motions $\boldsymbol{s} = [\alpha, \dot{\alpha}, \ddot{\alpha}, \dot{\xi}, \ddot{\xi}]$ and $\boldsymbol{x} = [x, y, z]$ the mesh coordinates.

Model	Inputs	Outputs
Autoencoder	$[\boldsymbol{Y}(\tau), \boldsymbol{x}] \in \mathbb{R}^{n_n \times (1+3)}$, $\hat{\boldsymbol{A}} \in \mathbb{R}^{n_{e_n} \times 3}$	$\boldsymbol{Y}(\tau) \in \mathbb{R}^{n_n \times 1}$
Quasi-steady model	$[\boldsymbol{s}(\tau), \boldsymbol{x}] \in \mathbb{R}^{n_n \times (5+3)}$, $\hat{\boldsymbol{A}} \in \mathbb{R}^{n_{e_n} \times 3}$	$\boldsymbol{Y}(\tau) \in \mathbb{R}^{n_n \times 1}$
Recurrent model	$[\boldsymbol{Y}(\tau - \mathrm{d}\tau), \boldsymbol{s}(\tau), \boldsymbol{x}] \in \mathbb{R}^{n_n \times (1+5+3)}$, $\hat{\boldsymbol{A}} \in \mathbb{R}^{n_{e_n} \times 3}$	$\boldsymbol{Y}(\tau) \in \mathbb{R}^{n_n \times 1}$

With the above considerations, to the best of our knowledge, our proposed framework for spatio-temporal simulations is novel on these fronts:

1. Data-driven spatio-temporal predictions on unstructured domains.
2. A multi-resolution autoencoder scheme, comprising cross-domain conservative interpolation, for model memory efficiency and information propagation across the mesh.
3. A time-marching scheme with long-term error considerations.
4. Building-block implementation to address different tasks: field reconstructions from sparse data, steady-state mappings and dynamic simulations.

This framework was implemented in `PyTorch 1.11`,[4] an optimised deep-learning library in `Python`, and `PyTorch Geometric`,[5] a graph neural-network library built on `PyTorch`.

4. Results

The test case is for the BSCW wing. Results herein reported answer the following questions:

1. Is the multi-mesh autoencoder scheme suitable for field predictions on large, unstructured domains?
2. Is the recurrent framework adequate for spatio-temporal simulation of moving bodies, capable of handling error accumulation?
3. How does the model behave to distinct response signals?

Before getting started with a comprehensive comparison between the two proposed modelling approaches, we assess the adequacy of the autoencoder generated in the first sequence of the modular implementation. The complete analysis is presented for $M_\infty = 0.74$. Results for the independently generated recurrent models at $M_\infty = 0.80$ and 0.84 are also provided. In this work, we opted for generating independent models for each Mach number of interest, as opposed to a single large model with Mach number as a flow-condition input. In that situation, the developed framework will be identical, with Mach number being an additional (global) input. However, the model embedding the new variable would be larger. This motivated the choice of different models per each Mach number. From here on, the results are at $M_\infty = 0.74$ unless otherwise stated.

4.1. Autoencoder analysis

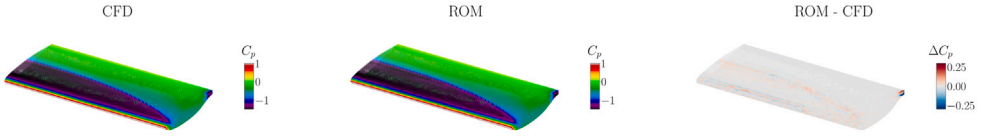
The results from the autoencoder, the first trained module in Fig. 6, are presented here. Details of the final architecture and optimisation strategy are provided in Appendix A.3. We assess the adequacy of the pure autoencoder model to reduce and recover back the C_p maps across the time history without general loss of information.

Figs. 8 and 9 illustrate snapshots of C_p encoded-decoded by the autoencoder from selected time instants of diverse response signals. Fig. 8 corresponds to a reconstructed snapshot from the training signal introduced in Fig. 2, evidencing a strong shock wave. Fig. 9, by contrast, compares a late instant from one of the validation responses. The 3D contour plots in Panels (a) present the C_p solutions for the ground truth (CFD), the model reconstruction (ROM) and the error (difference). Panels (b) provide a comparison of the C_p profiles at the three different cross sections of the wing indicated in Fig. 1 Panel (a), showcasing the variation of the flow field in span due to 3D effects. Panels (c) show the time evolution of the pitch and plunge input signals. The time instants where the contour plots are extracted from are highlighted on the time–history plots. These two snapshots in time evidence how the flow develops differently along the span (compare among the various cross sections) and among the various responses. This nonlinear behaviour makes the reconstruction and prediction tasks more challenging. Nevertheless, the reconstruction is good everywhere on the wing surface along the whole time period.

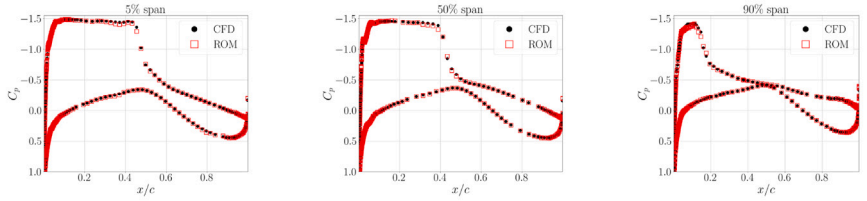
Panels (d) of the same Figures compare the resulting lift coefficient, between the ground truth (black line) and the prediction (red squares). Note that the forces are not directly predicted by the model but these are obtained by surface integration of the predicted pressure field history. The mean absolute error of the C_p distribution is also illustrated, defined as:

$$\varepsilon_{C_p}(\tau) = \sum_{i}^{n_n} |C_{p_{i,\mathrm{pred}}}(\tau) - C_{p_{i,\mathrm{ref}}}(\tau)| \tag{12}$$

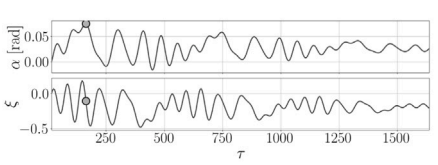
[4] https://pytorch.org/
[5] https://pytorch-geometric.readthedocs.io/en/latest/



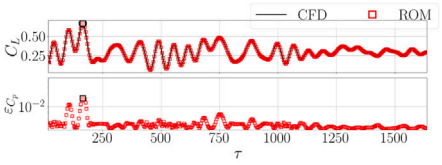
(a) Surface C_p comparison: CFD (left), ROM (middle) and the difference (right).



(b) C_p distributions at three different sections, model prediction (red squares) and CFD reference (black dots).

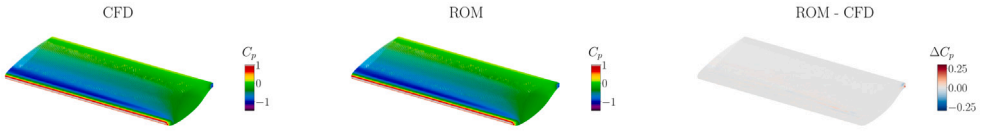


(c) Pitch and plunge motion signals.

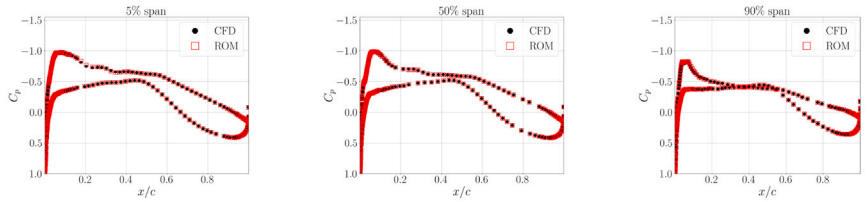


(d) Lift coefficient comparison and mean absolute error on C_p.

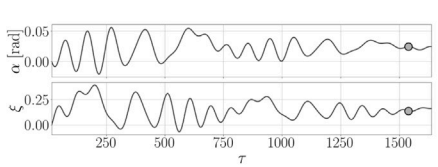
Fig. 8. Autoencoder reconstruction compared to CFD reference at $\tau = 165$ of the BSCW training signal 1. Worst reconstructed time instant. Upper surface. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)



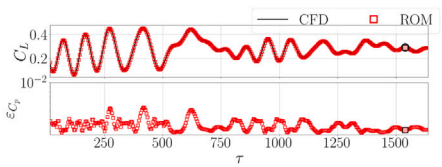
(a) Surface C_p comparison: CFD (left), ROM (middle) and the difference (right).



(b) C_p distributions at three different sections, model prediction (red squares) and CFD reference (black dots).



(c) Pitch and plunge motion signals.



(d) Lift coefficient comparison and mean absolute error on C_p.

Fig. 9. Autoencoder reconstruction compared to CFD reference at $\tau = 1540$ of the BSCW validation signal 1. Upper surface. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Table 4
Summary of the error analysis from the autoencoder reconstruction for the various signals for the BSCW test case.

Signal	ε_{C_p}			ε_{C_L} [%]		
	mean	std dev	worst	mean	std dev	worst
Training 1	$3.9{\cdot}10^{-3}$	$1.7{\cdot}10^{-3}$	$1.6{\cdot}10^{-2}$	0.4	0.2	1.5
Training 2	$3.8{\cdot}10^{-3}$	$1.5{\cdot}10^{-3}$	$1.5{\cdot}10^{-2}$	0.3	0.2	1.4
Validation 1	$3.5{\cdot}10^{-3}$	$4.9{\cdot}10^{-4}$	$5.4{\cdot}10^{-3}$	0.4	0.2	0.8
Validation 2	$4.3{\cdot}10^{-3}$	$2.1{\cdot}10^{-3}$	$1.7{\cdot}10^{-2}$	0.3	0.2	1.1
Validation 3	$3.9{\cdot}10^{-3}$	$6.9{\cdot}10^{-4}$	$5.7{\cdot}10^{-3}$	0.3	0.2	0.9
Validation 4	$4.2{\cdot}10^{-3}$	$1.8{\cdot}10^{-3}$	$1.2{\cdot}10^{-2}$	0.3	0.2	1.1
Validation 5	$3.6{\cdot}10^{-3}$	$1.8{\cdot}10^{-4}$	$3.9{\cdot}10^{-3}$	0.1	0.1	0.4
Validation 6	$4.2{\cdot}10^{-3}$	$1.5{\cdot}10^{-3}$	$1.2{\cdot}10^{-2}$	0.3	0.2	1.2

Table 5
Summary of the error analysis by the quasi-steady model prediction for the various signals for the BSCW test case.

Signal	ε_{C_p}			ε_{C_L} [%]		
	mean	std dev	worst	mean	std dev	worst
Training 1	$9.9{\cdot}10^{-3}$	$3.4{\cdot}10^{-3}$	$3.5{\cdot}10^{-2}$	1.7	1.4	9.0
Training 2	$9.4{\cdot}10^{-3}$	$3.9{\cdot}10^{-3}$	$3.7{\cdot}10^{-2}$	1.3	2.4	25.5
Validation 1	$8.3{\cdot}10^{-3}$	$1.3{\cdot}10^{-3}$	$1.3{\cdot}10^{-2}$	1.0	0.8	3.7
Validation 2	$1.1{\cdot}10^{-2}$	$5.9{\cdot}10^{-3}$	$4.1{\cdot}10^{-2}$	1.6	3.2	30.5
Validation 3	$1.1{\cdot}10^{-2}$	$2.1{\cdot}10^{-3}$	$1.6{\cdot}10^{-2}$	2.2	1.5	6.4
Validation 4	$1.0{\cdot}10^{-2}$	$5.7{\cdot}10^{-3}$	$3.8{\cdot}10^{-2}$	1.4	1.8	13.5
Validation 5	$8.2{\cdot}10^{-3}$	$6.7{\cdot}10^{-4}$	$1.1{\cdot}10^{-2}$	1.0	0.5	2.5
Validation 6	$1.0{\cdot}10^{-2}$	$2.5{\cdot}10^{-3}$	$2.3{\cdot}10^{-2}$	1.7	1.3	7.1

No normalisation was applied to avoid division by near-zero reference values. The loads are well predicted and the ε_{C_p} is consistently low throughout, confirming adequate data conversion onto a mesh 16 times coarser.

To complete the analysis, Table 4 reports a statistical summary of the ε_{C_p} Eq. (12) and the relative error on the C_L:

$$\varepsilon_{C_L}(\tau)\,[\%] = \frac{|C_{L\text{pred}}(\tau) - C_{L\text{ref}}(\tau)|}{|C_{L\text{ref}}(0)|} \cdot 100 \tag{13}$$

normalised by the initial value, i.e. the static condition, to avoid skewed errors due to division by near-zero value in instances when the lift coefficient inverts. The mean and standard deviation of the prediction error are reported for each response, as well as the worst instantaneous prediction. The average error is very low for either the C_p maps and the integrated forces. Furthermore, the low standard deviations and worst time steps denote consistent accuracy along the time period.

4.2. Quasi-steady modelling analysis

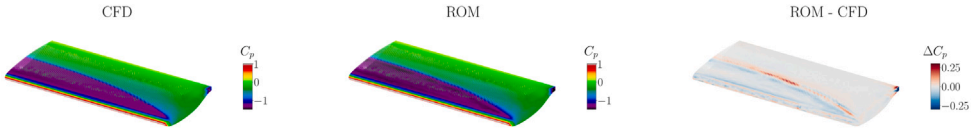
Upon demonstrating successful reconstruction performance by the autoencoder, we proceed with analysis of the quasi-steady GCN-MM-AE model results for the various forced-motion signals. Details of the quasi-steady encoder involved in the second module described in Section 3.3, which predicts the latent state of the autoencoder from the motion inputs, are reported in the last part of the Appendix.

Figs. 10 and 11 analyse the quasi-steady predictions for selected snapshots from two other validation signals. The C_p predictions closely match the reference CFD solution, with the shock wave location correctly captured. The accurate predictions obtained for the undamped Schroeder response (validation 6) in Fig. 10 are noteworthy since the model learned from signals with decaying amplitude. The quasi-steady model manages to capture the physics remarkably well, despite not accounting for the past flow solution. Observing Fig. 11 for validation response 2, another complex phenomenon becomes apparent: translation of the shock wave towards the leading edge and alternating across the upper and the bottom surfaces, known as shock-wave motion type C (Oswatitsch et al., 1976). This phenomenon is observed in multiple instances and it was found effectively captured by the model. However, at this snapshot, the model showcases a larger pressure gradient downstream of the shock wave. This discrepancy has minor implication on the resulting C_L, as seen in Panel (d), and fades away after a certain period. In conclusion, the quasi-steady implementation was found capable of capturing the complex flow features despite disregarding the past-state information.

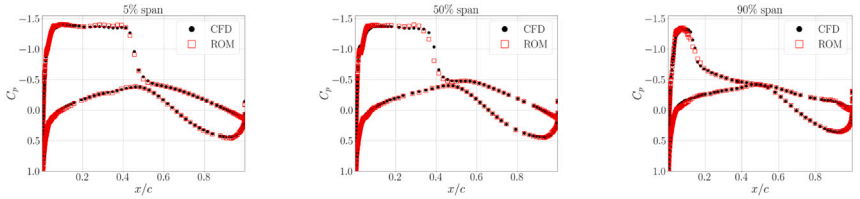
The statistical summary of the error metrics for the various response signals are reported in Table 5. The errors are larger compared to simply the reconstructions by the autoencoder, Table 4. This is expected since actual aerodynamic predictions of the moving wing are now addressed.

4.3. Recurrent modelling analysis

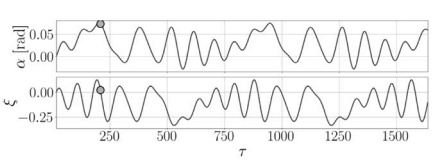
Details of the final recurrent GCN-MM-AE model architecture, corresponding to the second module in Fig. 6 Panel (b), as well as training strategy are reported in Table 11 of the Appendix A.4.



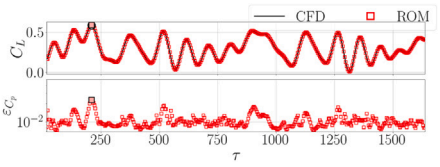
(a) Surface C_p comparison: CFD (left), ROM (middle) and the difference (right).



(b) C_p distributions at three different sections, model prediction (red squares) and CFD reference (black dots).

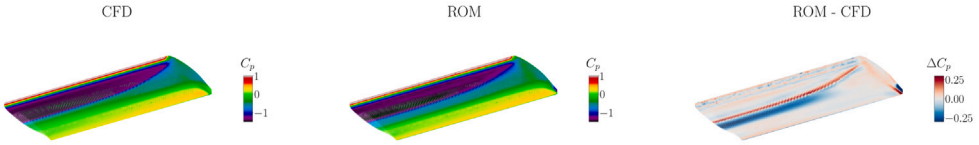


(c) Pitch and plunge motion signals.

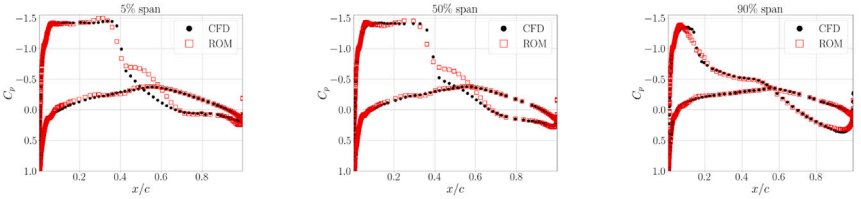


(d) Lift coefficient comparison and mean absolute error on C_p.

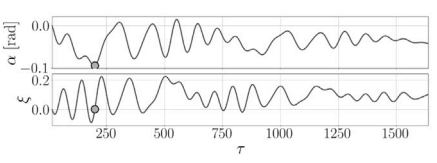
Fig. 10. Quasi-steady prediction compared to CFD reference at $\tau = 210$ of the BSCW validation signal 6. Upper surface. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)



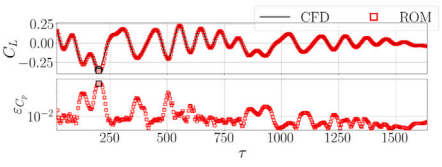
(a) Surface C_p comparison: CFD (left), ROM (middle) and the difference (right).



(b) C_p distributions at three different sections, model prediction (red squares) and CFD reference (black dots).



(c) Pitch and plunge motion signals.



(d) Lift coefficient comparison and mean absolute error on C_p.

Fig. 11. Quasi-steady prediction compared to CFD reference at $\tau = 200$ of BSCW validation signal 2. Worst predicted time instant. Lower surface. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Table 6
Summary of the error analysis by the recurrent model prediction for the various signals for the BSCW test case.

Signal	ϵ_{C_p}			ϵ_{C_L} [%]		
	mean	std dev	worst	mean	std dev	worst
Training 1	$9.0{\cdot}10^{-3}$	$3.9{\cdot}10^{-3}$	$5.1{\cdot}10^{-2}$	1.5	1.9	25.2
Training 2	$1.1{\cdot}10^{-2}$	$4.4{\cdot}10^{-3}$	$5.7{\cdot}10^{-2}$	1.8	3.0	27.5
Validation 1	$8.9{\cdot}10^{-3}$	$1.8{\cdot}10^{-3}$	$1.6{\cdot}10^{-2}$	2.1	1.7	13.4
Validation 2	$1.3{\cdot}10^{-2}$	$5.3{\cdot}10^{-3}$	$3.9{\cdot}10^{-2}$	2.2	2.7	22.6
Validation 3	$3.3{\cdot}10^{-2}$	$1.8{\cdot}10^{-2}$	$7.4{\cdot}10^{-2}$	10.8	7.4	29.5
Validation 4	$1.8{\cdot}10^{-2}$	$7.8{\cdot}10^{-3}$	$3.2{\cdot}10^{-2}$	2.8	1.8	8.6
Validation 5	$1.0{\cdot}10^{-2}$	$1.5{\cdot}10^{-3}$	$2.4{\cdot}10^{-2}$	1.1	1.3	12.4
Validation 6	$1.1{\cdot}10^{-2}$	$3.7{\cdot}10^{-3}$	$3.1{\cdot}10^{-2}$	2.7	2.1	13.6

Analyses of the predictions with the recurrent model on the same time snapshots are shown in Figs. 12 and 13. The C_p maps matched well even in these distinct aerodynamic conditions. In the undamped Schroeder case, Fig. 12, we found that the error is notably kept low along the whole time period even without decaying motion amplitudes. The integrated C_L and C_{M_y} were found accurate in consequence. Therefore, the issue of long-term error accumulation, critical in time-marching schemes, appears adequately handled by our model, proving suitability of the tailored training mechanisms adopted. Focusing on the second validation response analysis of Fig. 13, the error on the pressure gradient downstream of the shock wave observed with the quasi-steady model, Fig. 11, vanishes here. Nevertheless, both models manage to accurately predict the nonlinear variations and the 3D effects along the span correctly.

The statistical analysis of the results obtained with the recurrent model is reported in Table 6 for all the responses. Prediction accuracy is high in all cases, with the single-harmonic signal (validation 3) being worse than the others, as observed before. Nevertheless, both recurrent and quasi-steady models present similar error statistics. Remarkably, with either modelling approach, no significant performance degradation is observed between the validation and the training responses. This indicates that both models are not susceptible of over-fitting, a desired aspect when resorting to data-driven modelling.

4.4. Analysis on distinct response signals

Adequately learning the physics of the system is critical for a data-driven model, so that prediction accuracy is maintained even in distinct type of responses, property known as generalisation. This is essential to achieve a prediction model suitable for varying analyses. Figs. 14 and 15 report the response error by the quasi-steady (left) and the recurrent (right) models for two distinctive motion signals: Fig. 14 corresponds to uncoupled single-harmonic input signals for pitch and plunge (validation signal 3), resembling an undamped elastically suspended wing response; Fig. 15 is for a plunge only motion (signal 5), which is interesting because pitch usually exerts predominance on the aerodynamic response. These new signals provided further evidence about the reliability of our models. Each Panel reports, in ascending order, the two mode inputs, the comparison of the integrated C_L between prediction (red squares) and reference (black line), the relative error for the lift ϵ_{C_L} and the mean absolute error for the pressure distribution ϵ_{C_p}.

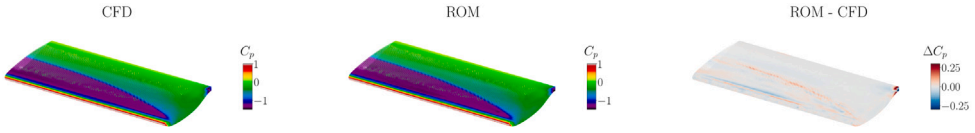
The magnitude of the errors were in general found similar between the two models, albeit slightly better with the quasi-steady case. However, two particularities are observed from the recurrent solutions:

1. For the single-harmonic response, from $\tau \sim 250$ to $\tau \sim 600$ a gradual degradation is observed with the recurrent predictions (upper Panels). However, this transitory error accumulation is stabilised thereupon.
2. For the plunge-only response, the recurrent solution present an initially large error but this is adequately rectified within a few time steps. The reason for this behaviour is unclear but this could be attributed to the GRU core managing to correct a bad initial prediction.

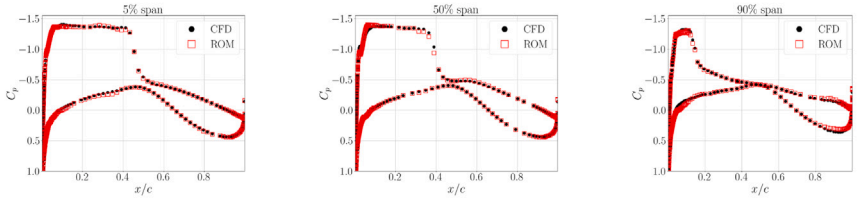
These phenomena are not observed with the quasi-steady model (left Panels). With this approach the physics are still well predicted, despite missing the recurrent feedback. Nonetheless, the performance by either approach is noteworthy considering that just four high-fidelity responses were employed to generate the models.

4.5. Analysis at $M_\infty = 0.80$

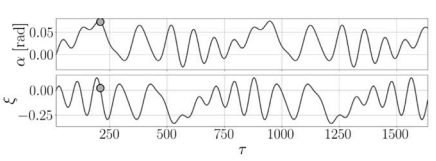
We are now interested in assessing the prediction performance of our framework for higher Mach numbers, where the shock-wave and boundary-layer interactions become gradually more complex. This section is for results at $M_\infty = 0.80$. Dedicated CFD responses at this new freestream condition were used to generate new models, following the same procedure described in the previous sections. For brevity, here we present the time-marching predictions obtained with the recurrent model Fig. 6 Panel (b). We selected the worst time steps from the various responses, which are illustrated in Figs. 16 and 17. Panel (a) is for the same validation signal as in the previous analyses while Panel (b) is for the uncoupled single harmonic signal. We observe how the physics have become more challenging to predict, with a more trapezoidal shock-wave line (as opposed to elliptic before) and fixed at 60% of the chord for most of the span. Nevertheless, the predictions are overall in good agreement and the forces are well predicted across the time history. The recurrent model is again able to keep error accumulation under control along the time history. There are, however, more evident



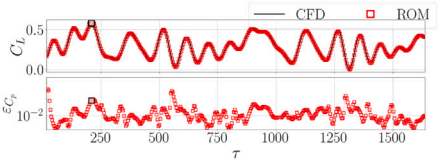
(a) Surface C_p comparison: CFD (left), ROM (middle) and the difference (right).



(b) C_p distributions at three different sections, model prediction (red squares) and CFD reference (black dots).

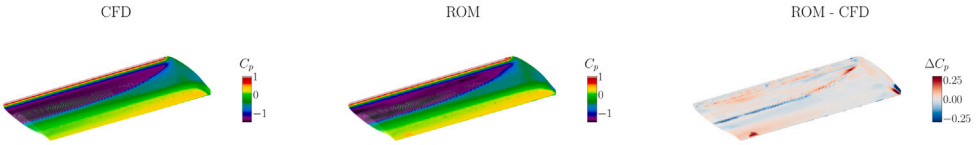


(c) Pitch and plunge motion signals.

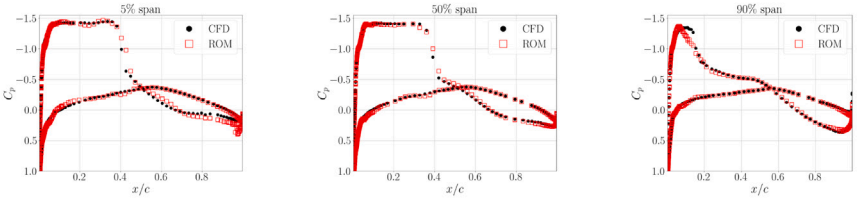


(d) Lift coefficient comparison and mean absolute error on C_p.

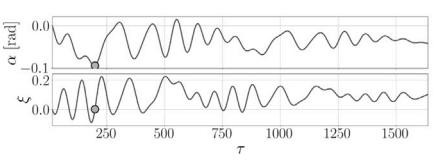
Fig. 12. Recurrent prediction compared to CFD reference at $\tau = 210$ of the BSCW validation signal 6. Upper surface. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)



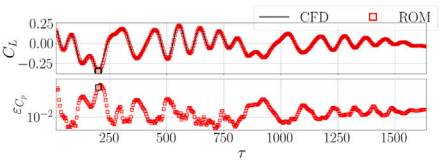
(a) Surface C_p comparison: CFD (left), ROM (middle) and the difference (right).



(b) C_p distributions at three different sections, model prediction (red squares) and CFD reference (black dots).

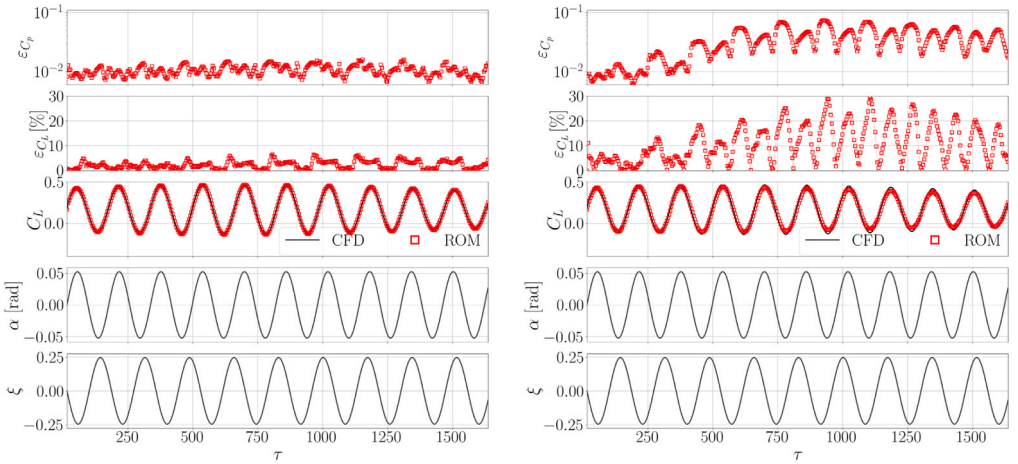


(c) Pitch and plunge motion signals.



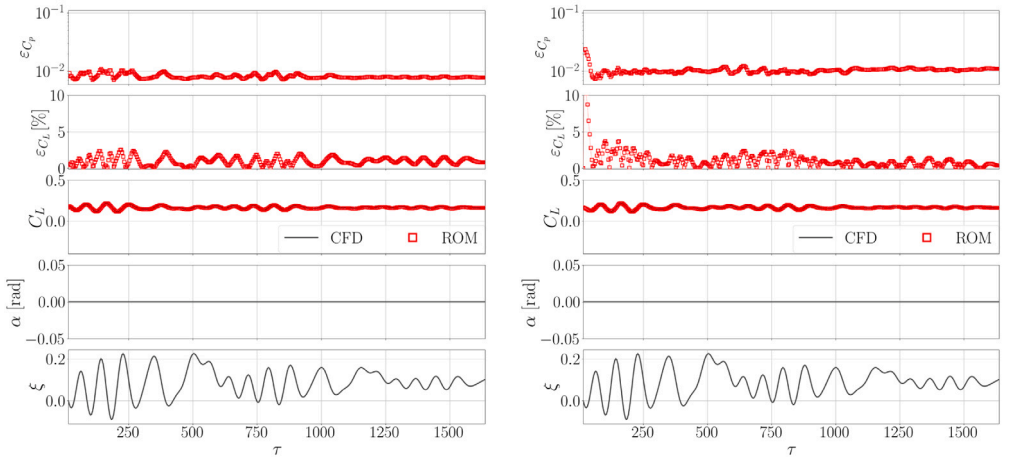
(d) Lift coefficient comparison and mean absolute error on C_p.

Fig. 13. Recurrent predictions compared to CFD reference at $\tau = 200$ of the BSCW validation signal 2. Worst predicted time instant. Lower surface. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)



(a) Quasi-steady model prediction. (b) Recurrent model prediction.

Fig. 14. Error history from the two ROMs for the single harmonic signal on the BSCW case.



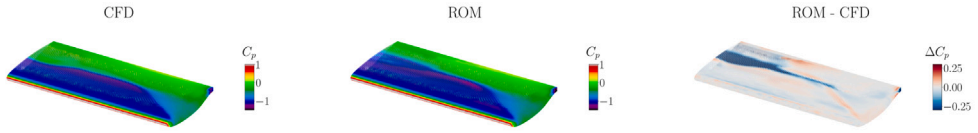
(a) Quasi-steady model prediction. (b) Recurrent model prediction.

Fig. 15. Error history from the two ROMs for the plunge-only signal on the BSCW case.

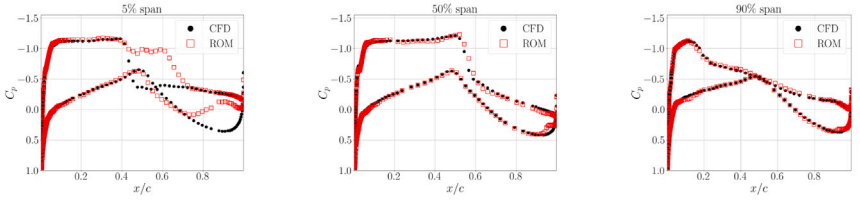
local discrepancies. In Fig. 16, the shock wave is predicted further downstream around the centreline of the wing. Nonetheless, this is a localised effect without repercussion on the overall loads and also, it is recovered subsequently. We know from Massegur et al. (2022) that at $M_\infty = 0.80$ a fluctuating shock-induced boundary layer separation and the successive re-attachment occur. At these conditions, intermittent local discrepancies are expected. In the single-harmonic case, Fig. 17, the solution is visibly better matched.

4.6. Analysis at $M_\infty = 0.84$

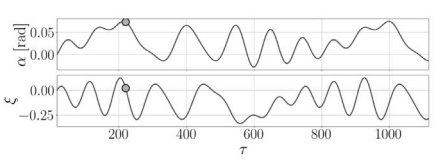
Finally, new models were generated at $M_\infty = 0.84$, with results shown in Figs. 18 and 19 for the same time instants as above. The physics are significantly more complex than at lower Mach numbers, featuring the most intense shock wave with an irregular shape along the span at around 45% of the chord. The strong shock causes a large and fluctuating boundary-layer separation region towards the trailing edge (Massegur et al., 2022). This phenomenon is evident from the CFD reference, left contour plot of Panels (a), with a rather irregular shock line and a flat C_p distribution downstream, indicating separated flow. We observe in the ΔC_p plots of the same Panels that the recurrent model manages to capture these phenomena adequately. However, further local discrepancies in the separation region downstream the shock wave are visible. This is more evident on the lower surface in Fig. 19. Note that CFD responses were solved using the one-equation SA turbulence model. This more basic model is regarded inadequate for separated flows, often causing poor solution convergence. As a result, seeing larger discrepancies in the separated regions of the wing are not surprising, since the reference SA-based solution is already uncertain in that region, to begin with. Nevertheless, and with



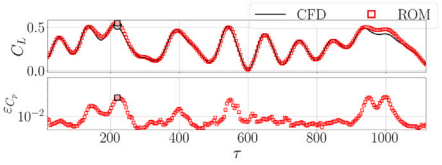
(a) Surface C_p comparison: CFD (left), ROM (middle) and the difference (right).



(b) C_p distributions at three different sections, model prediction (red squares) and CFD reference (black dots).

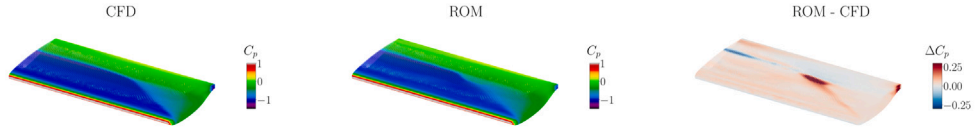


(c) Pitch and plunge motion signals.

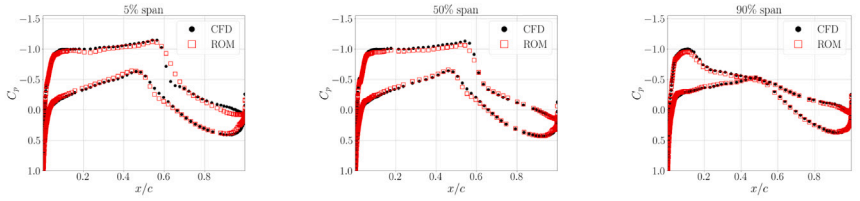


(d) Lift coefficient comparison and mean absolute error on C_p.

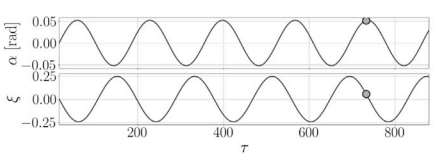
Fig. 16. Recurrent prediction compared to CFD reference at $M_\infty = 0.80$ and $\tau = 210$ of the BSCW validation signal 6. Worst predicted time instant. Upper surface. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)



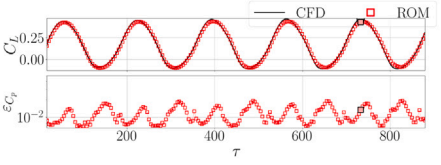
(a) Surface C_p comparison: CFD (left), ROM (middle) and the difference (right).



(b) C_p distributions at three different sections, model prediction (red squares) and CFD reference (black dots).

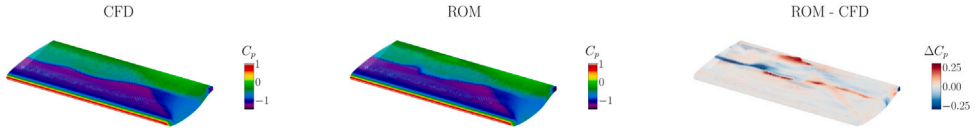


(c) Pitch and plunge motion signals.

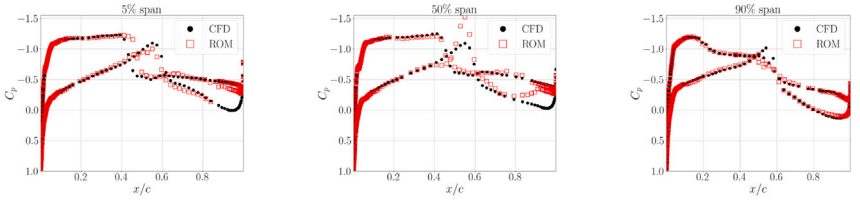


(d) Lift coefficient comparison and mean absolute error on C_p.

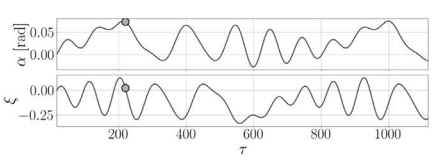
Fig. 17. Recurrent predictions compared to CFD reference at $M_\infty = 0.80$ and $\tau = 735$ of the BSCW validation signal 3. Upper surface. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)



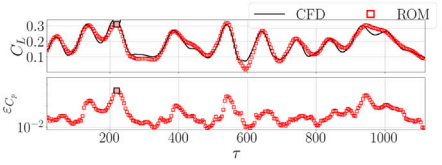
(a) Surface C_p comparison: CFD (left), ROM (middle) and the difference (right).



(b) C_p distributions at three different sections, model prediction (red squares) and CFD reference (black dots).

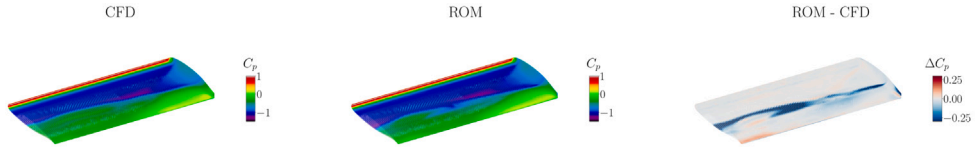


(c) Pitch and plunge motion signals.

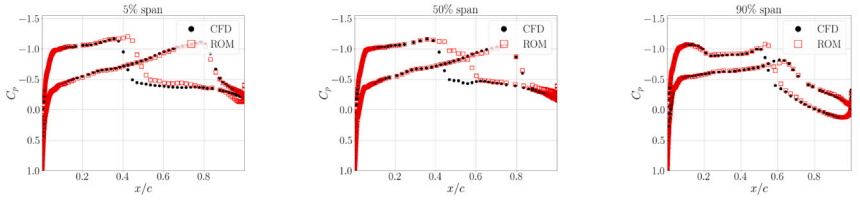


(d) Lift coefficient comparison and mean absolute error on C_p.

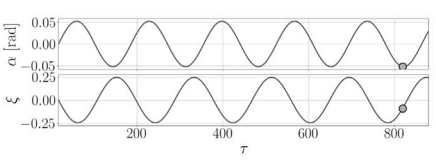
Fig. 18. Recurrent prediction compared to CFD reference at $M_\infty = 0.84$ and $\tau = 210$ of the BSCW validation signal 6. Upper surface. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)



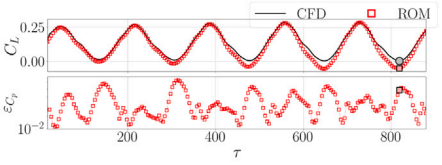
(a) Surface C_p comparison: CFD (left), ROM (middle) and the difference (right).



(b) C_p distributions at three different sections, model prediction (red squares) and CFD reference (black dots).



(c) Pitch and plunge motion signals.



(d) Lift coefficient comparison and mean absolute error on C_p.

Fig. 19. Recurrent predictions compared to CFD reference at $M_\infty = 0.84$ and $\tau = 820$ of the BSCW validation signal 3. Worst predicted time instant. Lower surface. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Table 7
Summary of the computing costs involved in forced-motion simulations. A multi-core cluster was used for the CFD solutions, while a standard GPU sufficed for the generation of the NN models.

Task	Computing cost
1 unsteady CFD response:	10,920 CPU-h
Training dataset of 4 CFD responses:	43,680 CPU-h
Autoencoder (module 1) training execution:	7.8 GPU-h
Quasi-steady model (module 2) training execution:	8.7 GPU-h
1 quasi-steady ROM response:	0.040 GPU-h
Recurrent model (module 2) training execution:	23.3 GPU-h
1 time-marching ROM response:	0.045 GPU-h

this in mind, our recurrent framework proves adequate to capture the distinct physics in different conditions and robust to error accumulation issues.

4.7. Computing cost analysis

Table 7 reports the computing costs required to generate each model and the significant savings compared to high-fidelity responses. CFD solutions were computed in a 280-core HPC, whereas our developed ROMs were generated with a conventional 8 GB `NVIDIA` GPU. Response signals were prescribed to $\tau = 1640$, i.e. 2 s of physical time. Refer to Tables 10 and 12 in the Appendix section for more details on the computing resources required by the various models. The saving in computational resources to complete a time-dependent simulation is over 99.9% with either model against high-fidelity CFD. Therefore, our models result increasingly more advantageous in any task requiring more than four simulations. Furthermore, the quasi-steady implementation involves lower memory usage, leading to gains of about 62% in training cost and up to 11% in new simulations compared to the recurrent counterpart, while maintaining similar performance accuracy.

4.8. Remarks

Two final remarks are made on the use of a classic technique such as Proper Orthogonal Decomposition (POD) and the applicability of the framework to more complex configurations.

POD method with Galerkin interpolation (Luchtenburg et al., 2009) compares to our quasi-steady implementation, since POD does not account for temporal terms. On the contrary, the time-dependent adaptation of POD is the Dynamic Mode Decomposition (DMD) method (Fonzi et al., 2020), which is a linearised projection of a dynamic system expressed in first-order form. DMD compares to the recurrent approach herein proposed. The well-known limitation with POD/DMD is that the linearised projection to the compressed state is performed via the eigenvector (or modes). In a previous contribution, we compared a convolutional autoencoder against POD (Massegur et al., 2023). The neural-net approach was found to significantly outperform the classic methods. Despite the POD being easier to implement, this did not compensate for the critically worse prediction accuracy. The same conclusions would apply here, where both the GNN and MM schemes embed nonlinear representations and projections, as opposed to the linearised treatment by the POD/DMD.

This work is centred around the BSCW wing, a common aeroelastic test case featuring a relatively simple wing geometry but highly complex flow dominated by dynamic shock waves and separated flow (Heeg and Chwalowski, 2017; Heeg et al., 2016). The proposed framework carries no limitations on the applicability to more complex geometry configurations and/or different flow conditions. In a separate study (Massegur and Da Ronch, 2024b), the proposed framework is used to predict unsteady aerodynamic loads around a next-generation fighter configuration (DLR-FFD) with highly swept delta wings. The flow is dominated by a system of vortices that build-up and break-down during dynamic manoeuvres at high angles of attack. The reader is invited to refer to Massegur and Da Ronch (2024b) for more details.

5. Conclusions

Reduced order modelling offers the ability to reduce the computational cost associated with unsteady, high-fidelity aerodynamic analysis. The work herein presented details a novel geometric-deep-learning autoencoder framework to predict unsteady aerodynamic loads around a three-dimensional wing geometry undergoing dynamic manoeuvres. The framework features:

1. Graph convolutional networks for unstructured meshes.
2. Multi-mesh autoencoder scheme to improve the computational efficiency and extract multi-scale features.
3. Time-marching scheme with long-term error considerations.
4. Building-block implementation to address different tasks, including coarse field reconstructions, quasi-steady and recurrent schemes.

The multi-resolution scheme is capable of operating on multiple mesh levels, up to 16 times coarser than the original domain. This approach extracts information contained in different spatial scales and significantly contributes towards enhancing the computational efficiency. On the time-dependent front, the critical challenges involved with time-marching schemes were addressed by embedding the GRU core, the two-module procedure and the training mechanisms. Furthermore, we used interchangeable blocks to evaluate and compare the performance between the quasi-steady and the recurrent frameworks. The model accuracy was found consistent throughout the time history, successfully addressing the risk of long-term error accumulation. The models were found to adequately capture the pressure fields in the presence of large gradients, e.g. from moving shock waves.

Related to the computational efficiency, the cost involved with the CFD generation of the training data must also be considered. Despite deep learning algorithms have reputation of requiring large amount of data, with our strategic choice of minimal data requirements, we demonstrated the adequacy of our models generated with just four training signals.

With a reduction of computational time of about 99.9% for a time-domain response, our framework provides nearly cost-free predictions compared with CFD when more than four responses are required. Hence, the more cases are analysed with the models, the larger the overall computational savings. Between the two approaches, the quasi-steady model was found still capable of capturing the unsteady nature of the physics with better computational efficiency, as it avoids the larger memory requirements by the GRU core in the recurrent option.

In its current development, the large cost to train the GRU-based architecture is attributed to:

1. The more intensive processing involved with the multiple time-marching stepping sequences described in Section 3.2.1.
2. The size of dataset time steps required to represent the complete design space resulted long to process with a single GPU, despite only four signals were used.

In tasks where the training data requirements must be further reduced due to computational resource limitations, a hybrid data-driven and physics-based contribution could be thought, consisting of introducing aerodynamics knowledge into the current method. However, fluid-dynamics based terms applicable to the aircraft surfaces are not straightforward and require proper consideration.

The developed aerodynamic model can be coupled to a structural model in a complete aeroelastic system for fast and reliable prediction of the flutter boundary. Note that our model architecture already leverages the mesh node coordinates as inputs and connectivity to embed a spatial mapping. As a result, the framework is already designed to deal with wing deformations to address aeroelastic analyses, and geometric changes to address design shape optimisation. In order for the model to execute prediction on varying wing shapes, the training dataset should be expanded to include additional samples of different geometries or deflected shapes. In this scenario, the graph (nodes and connectivity) embedded in our GNN-based framework would change for each different shape. This topic will be explored in a future work.

CRediT authorship contribution statement

David Massegur: Writing – review & editing, Writing – original draft, Validation, Software, Project administration, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Andrea Da Ronch:** Writing – review & editing, Visualization, Validation, Supervision, Conceptualization.

Declaration of competing interest

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

Acknowledgements

The authors acknowledge the use of the IRIDIS High-Performance Computing Facility, and associated support services at the University of Southampton, in the completion of this work.

Funding

This research received no external funding.

Appendix

A.1. Training and validation signals

Table 8 defines the first mode (frequency and amplitude) of the modified Schroeder signals used for the generation and validation of the various ROMs to simulate the BSCW wing subject to pitch and plunge motion. In particular, we adopted four damped

Table 8
Definition of the time-dependent signals used for training and validation. The starting first-mode amplitude and frequency for the pitch α and plunge ξ forced motions are detailed. Signals correspond to damped Schroeder Eq. (1) unless otherwise stated. $<,>$ indicate, respectively, a negative or positive sign of the state along the signal history.

Signals	ω_α [Hz]	$a_\alpha(t_0)$ [deg]	ω_ξ [Hz]	$a_\xi(t_0)$ [-]	Description
Training 1	1.40	0.80	−1.60	−0.098	$\alpha > 0$, $\xi < 0$, high freq.
Training 2	1.40	−0.80	−1.60	0.098	$\alpha < 0$, $\xi > 0$, high freq.
Training 3	0.65	1.00	−0.85	−0.123	$\alpha > 0$, $\xi < 0$, low freq.
Training 4	0.65	−1.00	−0.85	0.123	$\alpha < 0$, $\xi > 0$, low freq.
Validation 1	−1.05	0.70	1.11	0.074	$\alpha > 0$, $\xi > 0$, mid freq.
Validation 2	1.10	−1.00	−1.25	0.049	$\alpha < 0$, $\xi > 0$, mid freq.
Validation 3	5.10	3.00	4.80	−0.246	single harmonics
Validation 4	1.10	−1.00	0.00	0.000	pitch only
Validation 5	0.00	0.00	−1.25	0.049	plunge only
Validation 6	1.10	0.75	−1.20	−0.059	undamped Schroeder

Table 9
Architecture for the autoencoder (GCN-MM-AE) adopted in the first module of the recurrent framework, which compresses and reconstructs the C_p field.

Layer	Dimension	Operation	Kernel
Input: C_{p_i}, $\boldsymbol{x}_i$, $\hat{\boldsymbol{A}}_0$	$m \times 86840 \times 4$		
Encoder:			
GCN Enc0.1	$m \times 86840 \times 72$	Eq. (6)	$4 \times 72 + 72$
PReLU Enc0.1	$m \times 86840 \times 72$	Eq. (7)	72
GCN Enc0.2	$m \times 86840 \times 144$	Eq. (6)	$72 \times 144 + 144$
PReLU Enc0.2	$m \times 86840 \times 144$	Eq. (7)	144
MM $\hat{\boldsymbol{A}}_0 \leftarrow .2cm\, \hat{\boldsymbol{A}}_1$	$m \times 5427 \times 144$	Eq. (8)	
GCN Enc1.1	$m \times 5427 \times 144$	Eq. (6)	$144 \times 144 + 144$
PReLU Enc1.1	$m \times 5427 \times 144$	Eq. (7)	144
GCN Enc1.2	$m \times 5427 \times 288$	Eq. (6)	$144 \times 288 + 288$
PReLU Enc1.2	$m \times 5427 \times 288$	Eq. (7)	288
Decoder:			
GCN Dec1.1	$m \times 5427 \times 144$	Eq. (6)	$288 \times 144 + 144$
PReLU Dec1.1	$m \times 5427 \times 144$	Eq. (7)	144
GCN Dec1.2	$m \times 5427 \times 144$	Eq. (6)	$144 \times 144 + 144$
PReLU Dec1.2	$m \times 5427 \times 144$	Eq. (7)	144
MM $\hat{\boldsymbol{A}}_1 \leftarrow .2cm\, \hat{\boldsymbol{A}}_0$	$m \times 86840 \times 144$	Eq. (8)	
GCN Dec0.1	$m \times 86840 \times 72$	Eq. (6)	$144 \times 72 + 72$
PReLU Dec0.1	$m \times 86840 \times 72$	Eq. (7)	72
GCN Dec0.2	$m \times 86840 \times 72$	Eq. (6)	$72 \times 72 + 72$
PReLU Dec0.2	$m \times 86840 \times 72$	Eq. (7)	72
Repeat: $\mathbf{y}_i \leftarrow .2cm\, [\mathbf{y}_i]$	$m \times 86840 \times 72 \times 1$		
GCN Dec0.3	$m \times 86840 \times 72 \times 1$	Eq. (6)	$72 \times 72 \times 1 + 72 \times 1$
PReLU Dec0.3	$m \times 86840 \times 72 \times 1$	Eq. (7)	72×1
GCN Dec0.4	$m \times 86840 \times 1 \times 1$	Eq. (6)	$72 \times 1 \times 1 + 1 \times 1$
Output: C_{p_i}	$m \times 86840 \times 1$	Eq. (10)	

Schroeder-based signals for the training, based on Eq. (1). The two extremes of representative structural main frequencies were chosen for the frequency values of the training signals. Also, values were assigned different between the two modes to ensure uncoupled aerodynamic response to pitch and plunge. Amplitudes were chosen to ensure sufficiently large displacements and subsequently capturing a representative amplitude value range given the prescribed damping of the modified Schroeder. Various signal types were prescribed for validation purposes, substantially different from those used for training to prove the capability of our aerodynamic models to various responses. The single-harmonic signal is defined by just the first mode in Eq. (1), whereas the original (undamped) Schroeder is obtained by assigning $a(t_{\text{end}}) = a(t_0)$ in Eq. (2). CFD responses were solved for three different Mach conditions $M_\infty = [0.74, 0.80, 0.84]$, and fixed $Re = 4.49 \cdot 10^6$, $q_\infty = 8000$ Pa and static $\alpha_\infty = 0$ deg.

A.2. Gated recurrent unit formulation

The gated recurrent unit (GRU) module was developed by Cho et al. (2014). RNNs are designed to embed past-history states, i.e. carry memory over, towards the subsequent prediction steps. A well known issue with vanilla RNN is the numerical instability from vanishing or exploding gradients in long-term forecasting sequences. To alleviate this issue, long short-term (LSTM) and transformer networks have become the state of the art in time-series tasks. However, these units are subject to large memory requirements due to disproportionate number of model parameters involved. In this context, the GRU option represents a compromise by attempting to address the training gradient issues with a simpler architecture which requires less memory resources.

Table 10
Training strategy adopted to generate the autoencoder (GCN-MM-AE) model, first module of the sequential framework.

Parameter	Value
Trainable parameters	158,185
Training signals	4
Time-step samples	1588
Training set (75%)	1191
Batch size	1
Training epochs	75
Loss function	MSE
Optimiser	Adam
Learning rate	0.003
Learning rate schedule	0.333/20 epochs
GPU machine	NVIDIA GeForce RTX 2070
Training time	7.8 h

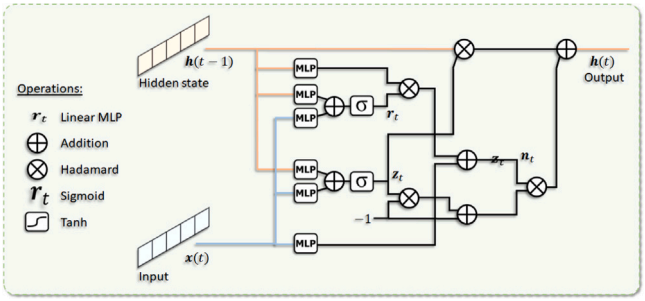


Fig. A.20. Diagram of the GRU operator.

GRU embeds a hidden state $\boldsymbol{h}_{t-1}$ with the input $\boldsymbol{x}_t$ to compute a new hidden state $\boldsymbol{h}_t$. The idea is to introduce two gates: the update gate $\boldsymbol{z}_t$ and the reset gate $\boldsymbol{r}_t$.[6] The update gate deals with the long-term memory and the reset gate handles the short-term one. In summary, the reset gate embeds the input and the previous hidden state to produce a hidden state candidate. Next, the update gate combines the previous hidden state with the new candidate to generate the new hidden state. As a result, the GRU operator is defined as:

$$\begin{aligned}
\boldsymbol{r}_t &= \sigma\left(\boldsymbol{W}_{ir}\boldsymbol{x}_t + \boldsymbol{b}_{ir} + \boldsymbol{W}_{hr}\boldsymbol{h}_{t-1} + \boldsymbol{b}_{hr}\right)\\
\boldsymbol{z}_t &= \sigma\left(\boldsymbol{W}_{iz}\boldsymbol{x}_t + \boldsymbol{b}_{iz} + \boldsymbol{W}_{hz}\boldsymbol{h}_{t-1} + \boldsymbol{b}_{hz}\right)\\
\boldsymbol{n}_t &= \tanh\left(\boldsymbol{W}_{in}\boldsymbol{x}_t + \boldsymbol{b}_{in} + \boldsymbol{r}_t \circ \left(\boldsymbol{W}_{hn}\boldsymbol{x}_t + \boldsymbol{b}_{hn}\right)\right)\\
\boldsymbol{h}_t &= \left(1 - \boldsymbol{z}_t\right) \circ \boldsymbol{n}_t + \boldsymbol{z}_t \circ \boldsymbol{h}_{t-1}
\end{aligned} \tag{A.1}$$

The six different pairs of $\boldsymbol{W}$ and $\boldsymbol{b}$ are the weights of the respective linear operators. The Hadamard (element-wise) product is denoted by $\circ$ symbol and $\sigma(x) = \frac{1}{1+e^{-x}}$ is the sigmoid function. Subscripts t and $t-1$ correspond to the current and previous time steps. Fig. A.20 illustrates the workflow of operations involved in the GRU unit.

A.3. Autoencoder architecture

Table 9 reports the final model architecture adopted for the pure autoencoder 4.1, i.e. the first module in Fig. 6. The various GCN and MM blocks for the encoder and decoder branches are evidenced. The PReLU nonlinear activation function Eq. (7) was applied at the output of each GCN layer, except for the last layer, which outputs the target prediction. The tensor dimension at each layer output is also reported in the Table, arranged in $m \times n_{\text{mml}} \times c_l$, with m the batch size, n_{mml} the grid size of the corresponding MM level and c_l the state size (or channels). The operation involved in each layer and the resulting kernel size (number of learnable parameters) are also included for clarity.

Table 10 provides further details regarding the training and optimisation strategies. We opted for the first-order gradient-based algorithm Adam (Kingma and Ba, 2015) as optimisation algorithm and the mean squared error (MSE) as loss function (Bickel and Doksum, 2015). As per recommended practice, the input data were standardised with the respective mean values and normalised with the standard deviation for a more efficient training process.

[6] https://www.analyticsvidhya.com/blog/2021/03/introduction-to-gated-recurrent-unit-gru/

Table 11
Architecture of the network branches added in the second module of the recurrent GCN-MM-AE model. This is combined with the autoencoder of Table 9 to compute the time-marching solution.

Layer	Dimension	Operation	Kernel
Input: $[\alpha, \dot{\alpha}, \ddot{\alpha}, \dot{\xi}, \ddot{\xi}]$, $\boldsymbol{x}_i$, $\hat{\boldsymbol{A}}_0$	$m \times 86840 \times 8$		
Encoder 2:			
GCN Enc2.1	$m \times 86840 \times 72$	Eq. (6)	$8 \times 72 + 72$
PReLU Enc2.1	$m \times 86840 \times 72$	Eq. (7)	72
GCN Enc2.2	$m \times 86840 \times 144$	Eq. (6)	$72 \times 144 + 144$
PReLU Enc2.2	$m \times 86840 \times 144$	Eq. (7)	144
MM $\hat{\boldsymbol{A}}_0 \leftarrow .2cm\, \hat{\boldsymbol{A}}_1$	$m \times 5427 \times 144$	Eq. (8)	
GCN Enc3.1	$m \times 5427 \times 144$	Eq. (6)	$144 \times 144 + 144$
PReLU Enc3.1	$m \times 5427 \times 144$	Eq. (7)	144
GCN Enc3.2	$m \times 5427 \times 288$	Eq. (6)	$144 \times 288 + 288$
PReLU Enc3.2	$m \times 5427 \times 288$	Eq. (7)	288
GRU	$m \times 5427 \times 288$	Eq. (A.1)	$288 \times 288 \times 6 + 288 \times 6$

Table 12
Training strategy to generate the recurrent GCN-MM-AE model, second module of the sequential framework.

Parameter	Value
Trainable parameters	$(158,185) + 573,840$
Training signals	4
Simulation dt	0.005 s
Training sequence length k	3 time steps
Time-step samples	788
Training set (75%)	591
Batch size	1
Training epochs	150
Loss function	MSE
Optimiser	Adam
Learning rate	0.003
Learning rate schedule	0.333/40 epochs
GPU machine	NVIDIA GeForce RTX 2070
Training time	23.3 h

A.4. Recurrent model architecture

The autoencoder architecture detailed above was subsequently frozen and employed in both quasi-steady and the recurrent models in the two-module training sequence, as described in Sections 3.2 and 3.3. Table 11 completes the recurrent GCN-MM-AE model architecture, this is the second encoder and the recurrent unit in Fig. 6. We observe that the single GRU core employs 78% of the memory resource, with around half a million parameters.

The resulting model size for the recurrent implementation significantly impacts on the training cost, as evidenced in Table 12, which provides the information on the strategy adopted for the second module of the sequential training process. By contrast, the quasi-steady implementation adopted just a second encoder to embed the input motion, without the GRU core. As a result, the number of additional parameters was reduced from 573,840 to 74,448, which was convenient as the optimisation could be completed in just 33% of the time compared to the recurrent counterpart.

Data availability

Data will be made available on request.

References

Anderson, J., 2016. Fundamentals of Aerodynamics. McGraw-Hill Education, URL https://books.google.co.uk/books?id=D1ZojgEACAAJ.

Beck, A., Kurz, M., 2020. A perspective on machine learning methods in turbulence modelling. URL http://arxiv.org/abs/2010.12226.

Bickel, P., Doksum, K., 2015. Mathematical Statistics: Basic Ideas and Selected Topics, Volume I, Second Edition. In: Chapman & Hall/CRC Texts in Statistical Science, CRC Press, URL https://books.google.co.uk/books?id=y5i9BwAAQBAJ.

Bronstein, M.M., Bruna, J., Cohen, T., Veličković, P., 2021. Geometric deep learning grids, groups, graphs, geodesics, and gauges. URL https://arxiv.org/abs/2104.13478.

Bronstein, M.M., Bruna, J., Lecun, Y., Szlam, A., Vandergheynst, P., 2017. Geometric deep learning: Going beyond Euclidean data. IEEE Signal Process. Mag. 34 (4), 18–42. http://dx.doi.org/10.1109/MSP.2017.2693418.

Brunton, S.L., Kutz, J.N., 2019. Data-driven science and engineering: Machine learning, dynamical systems, and control, first ed. Cambridge University Press, USA.

Brunton, S.L., Noack, B.R., Koumoutsakos, P., 2020. Machine learning for fluid mechanics. Annu. Rev. Fluid Mech. 52, 477–508. http://dx.doi.org/10.1146/annurev-fluid-010719-060214.

Cho, K., van Merrienboer, B., Bahdanau, D., Bengio, Y., 2014. On the properties of neural machine translation: Encoder-decoder approaches. URL http://arxiv.org/abs/1409.1259.

Chung, J., Gulcehre, C., Cho, K., 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv URL https://arxiv.org/abs/1412.3555.

Da Ronch, A., Ghoreyshi, M., Badcock, K.J., Görtz, S., Widhalm, M., Dwight, s.R.P., Campobasso, M.S., 2013. Linear frequency domain and harmonic balance predictions of dynamic derivatives. J. Aircr. 50 (3), http://dx.doi.org/10.2514/1.C031674.

Font, B., Weymouth, G.D., Nguyen, V.-t., Tutty, O.R., 2021. Deep learning of the spanwise-averaged Navier – Stokes equations. J. Comput. Phys. 434, 110199. http://dx.doi.org/10.1016/j.jcp.2021.110199.

Fonzi, N., Brunton, S.L., Fasel, U., 2020. Data-driven nonlinear aeroelastic models of morphing wings for control. Proc. R. Soc. A: Math. Phys. Eng. Sci. 476 (2239), 20200079. http://dx.doi.org/10.1098/rspa.2020.0079, URL https://royalsocietypublishing.org/doi/abs/10.1098/rspa.2020.0079.

Glaz, B., Liu, L., Friedmann, P.P., 2010. Reduced-order nonlinear unsteady aerodynamic modeling using a surrogate-based recurrence framework. AIAA J. 48 (10), 2418–2429. http://dx.doi.org/10.2514/1.J050471.

Goodfellow, I., Bengio, Y., Courville, A., 2016. Deep Learning. MIT Press, http://www.deeplearningbook.org.

Graves, A., 2013. Generating sequences with recurrent neural networks. URL http://arxiv.org/abs/1308.0850.

Han, X., Gao, H., Pfaff, T., Wang, J.-X., Liu, L.-P., 2022. Predicting physics in mesh-reduced space with temporal attention. URL http://arxiv.org/abs/2201.09113.

He, K., Zhang, X., Ren, S., Sun, J., 2015. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. URL http://arxiv.org/abs/1502.01852.

Heeg, J., Chwalowski, P., 2017. Investigating the transonic flutter boundary of the benchmark supercritical wing. In: 58th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference. p. 0191.

Heeg, J., Chwalowski, P., Raveh, D.E., Dalenbring, M.J., Jirasek, A., 2015. Plans and example results for the 2nd AIAA aeroelastic prediction workshop. In: 56th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference. p. 0437.

Heeg, J., Chwalowski, P., Raveh, D.E., Jirasek, A., Dalenbring, M., 2016. Overview and data comparisons from the 2nd aeroelastic prediction workshop. In: 34th AIAA Applied Aerodynamics Conference. p. 3121.

Hines, D., Bekemeyer, P., 2023. Graph neural networks for the prediction of aircraft surface pressure distributions. Aerosp. Sci. Technol. 137, http://dx.doi.org/10.1016/j.ast.2023.108268.

Hinton, G.E., Salakhutdinov, R.R., 2006. Reducing the Dimensionality of Data with Neural Networks. Science 313 (5786), 502–504. http://dx.doi.org/10.1126/science.1129198.

Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural computation 9, 1735–1780. http://dx.doi.org/10.1162/neco.1997.9.8.1735.

Joldes, G.R., Chowdhury, H.A., Wittek, A., Doyle, B., Miller, K., 2015. Modified moving least squares with polynomial bases for scattered data approximation. Appl. Math. Comput. 266, 893–902. http://dx.doi.org/10.1016/j.amc.2015.05.150, URL https://www.sciencedirect.com/science/article/pii/S0096300315007924.

Kingma, D.P., Ba, J.L., 2015. Adam: A method for stochastic optimization. 3rd Int. Conf. Learn. Represent., ICLR 2015 -Conf. Track Proc. 1–15.

Kipf, T.N., Welling, M., 2017. Semi-supervised classification with graph convolutional networks. 5th Int. Conf. Learn. Represent. ICLR 2017 - Conf. Track Proc. 1–14.

Lentz, J.J., Leek, M.R., 2001. Psychophysical estimates of cochlear phase response: Masking by harmonic complexes. JARO - J. Assoc. Res. Otolaryngol. 2 (4), 408–422. http://dx.doi.org/10.1007/s101620010045.

Liu, Y., Ponce, C., Brunton, S.L., Kutz, J.N., 2023. Multiresolution convolutional autoencoders. J. Comput. Phys. 474, http://dx.doi.org/10.1016/j.jcp.2022.111801.

Luchtenburg, D.M., Noack, B.R., Schlegel, M., 2009. An introduction to the POD Galerkin method for fluid flows with analytical examples and MATLAB source codes. Technical Report, Berlin Institute of Technology, Department for Fluid Dynamics and Engineering Acoustics, Berlin.

Mannarino, A., Mantegazza, P., 2014. Nonlinear aeroelastic reduced order modeling by recurrent neural networks. J. Fluids Struct. 48, 103–121. http://dx.doi.org/10.1016/j.jfluidstructs.2014.02.016.

Massegur, D., Clifford, D., Da Ronch, A., Lombardi, R., Panzeri, M., 2023. Low-dimensional models for aerofoil icing predictions. Aerosp. J. 10 (5), 444. http://dx.doi.org/10.3390/aerospace10050444, URL https://www.mdpi.com/2226-4310/10/5/444.

Massegur, D., Da Ronch, A., 2024a. Graph convolutional multi-mesh autoencoder for steady transonic aircraft aerodynamics. Mach. Learn.: Sci. Technol. 5 (2), 025006. http://dx.doi.org/10.1088/2632-2153/ad36ad, URL https://iopscience.iop.org/article/10.1088/2632-2153/ad36ad.

Massegur, D., Da Ronch, A., 2024b. Recurrent geometric deep learning for aerodynamic prediction of the future fighter demonstrator in dynamic manoeuvres. In: AIAA Aviation 2024 Forum. American Institute of Aeronautics and Astronautics, Las Vegas, Nevada, http://dx.doi.org/10.2514/6.2024-4067, URL https://arc.aiaa.org/doi/10.2514/6.2024-4067.

Massegur, D., Immordino, G., Da Ronch, A., Righi, M., Düzel, S., Anderegg, D., Soukhmane, I., 2022. ROM-Based Uncertainties Quantification of Flutter Speed Prediction of the BSCW Wing. In: AIAA SCITECH 2022 Forum. American Institute of Aeronautics and Astronautics, Reston, Virginia, http://dx.doi.org/10.2514/6.2022-0179, URL https://arc.aiaa.org/doi/10.2514/6.2022-0179.

McCormick, S.F., 1987. Multigrid Methods. Society for Industrial and Applied Mathematics, http://dx.doi.org/10.1137/1.9781611971057,

Morimoto, M., Fukami, K., Zhang, K., Nair, A.G., Fukagata, K., 2021. Convolutional neural networks for fluid flow analysis: toward effective metamodeling and low dimensionalization. Theor. Comput. Fluid Dyn. 35 (5), 633–658. http://dx.doi.org/10.1007/s00162-021-00580-0.

Ogoke, F., Meidani, K., Hashemi, A., Farimani, A.B., 2021. Graph convolutional networks applied to unstructured flow field data. Mach. Learn.: Sci. Technol. 2 (4), http://dx.doi.org/10.1088/2632-2153/ac1fc9.

Oswatitsch, K., Rues, D., International congress of theoretical and applied mechanics and International Union of Theoretical and Applied Mechanics, 1976. Symposium Transsonicum II: Symposium, Göttingen, September 8-13, 1975. IUTAM symposium, Springer, URL https://books.google.co.uk/books?id=JhT4zQEACAAJ.

Park, K.H., Jun, S.O., Baek, S.M., Cho, M.H., Yee, K.J., Lee, D.H., 2013. Reduced-order model with an artificial neural network for aerostructural design optimization. J. Aircr. 50 (4), 1106–1116. http://dx.doi.org/10.2514/1.C032062.

Pfaff, T., Fortunato, M., Sanchez-Gonzalez, A., Battaglia, P.W., 2020. Learning Mesh-Based Simulation with Graph Networks. URL http://arxiv.org/abs/2010.03409.

Pope, S.B., 2000. Turbulent Flows. Cambridge University Press.

Quaranta, G., Masarati, P., Mantegazza, P., 2005. A Conservative Mesh-Free Approach For Fluid-Structure Interface Problems. In: CIMNE (Ed.), Int. Conf. on Computational Methods for Coupled Problems in Science and Engineering, Coupled Problems 2005 - ECCOMAS. Barcelona, pp. 1–22.

Raveh, D.E., 2007. CFD-based models of aerodynamic gust response. J. Aircr. 44 (3), 888–897. http://dx.doi.org/10.2514/1.25498.

Ribau, A.M., Gonçalves, N.D., Ferrás, L.L., Afonso, A.M., 2021. Flow structures identification through proper orthogonal decomposition: The flow around two distinct cylinders. Fluids 6 (11), http://dx.doi.org/10.3390/fluids6110384.

Righi, M., Greco, P., Da Ronch, A., 2021. Uncertainties Quantification of CFD-Based Flutter Prediction. In: AIAA SCITECH 2021 Forum. pp. 1–14. http://dx.doi.org/10.2514/6.2021-1038.

Rozov, V., Breitsamter, C., 2021. Data-driven prediction of unsteady pressure distributions based on deep learning. J. Fluids Struct. 104, http://dx.doi.org/10.1016/j.jfluidstructs.2021.103316.

Sahinkaya, M.N., Cole, M.O.T., Burrows, C.R., 2002. On the use of schroeder phased harmonic sequences in multi-frequency vibration control of flexible rotor/magnetic bearing systems. Proc. Int. Symp. Magn. Bear. 217–222.

Silva, W.A., 1993. Application of nonlinear systems theory to transonic unsteady aerodynamic responses. J. Aircr. 30 (5), 660–668. http://dx.doi.org/10.2514/3.46395.

Smith, W.A., 1990. Multigrid solution of transonic flow on unstructured grids. In: Baysal, O. (Ed.), Recent Advances and Applications in Computational Fluid Dynamics. Proceedings of the ASME Winter Annual Meeting, Hanover.

Sureshbabu, S., Tejero, F., Sanchez-Moreno, F., MacManus, D.G., Sheaf, C., 2023. Deep-learning methods for non-linear transonic flow-field prediction. In: AIAA AVIATION 2023 Forum. American Institute of Aeronautics and Astronautics, San Diego, California, http://dx.doi.org/10.2514/6.2023-3719, URL https://arc.aiaa.org/doi/10.2514/6.2023-3719.

Thuerey, N., Holl, P., Mueller, M., Schnell, P., Trost, F., Um, K., 2021. Physics-based Deep Learning. WWW, URL https://physicsbaseddeeplearning.org.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I., 2017. Attention is all you need. URL http://arxiv.org/abs/1706.03762.

Wang, Q., Cesnik, C.E., Fidkowski, K., 2020. Multivariate Recurrent Neural Network Models for Scalar and Distribution Predictions in Unsteady Aerodynamics. In: AIAA Scitech 2020 Forum. American Institute of Aeronautics and Astronautics, Reston, Virginia, http://dx.doi.org/10.2514/6.2020-1533, URL https://arc.aiaa.org/doi/10.2514/6.2020-1533.

Wojtczak, M., Oxenham, A.J., 2009. On- and off-frequency forward masking by schroeder-phase complexes. JARO - J. Assoc. Res. Otolaryngol. 10 (4), 595–607. http://dx.doi.org/10.1007/s10162-009-0180-0.