An application of neural networks to the prediction of aerodynamic coefficients of aerofoils and wings

Kensley Balla, Ruben Sevilla, Oubay Hassan*, Kenneth Morgan

Zienkiewicz Centre for Computational Engineering,
Faculty of Science and Engineering, Swansea University,
Bay Campus, SA1 8EN, Wales, United Kingdom.

Abstract

This work proposes a novel multi-output neural network for the prediction of aerodynamic coefficients of aerofoils in two dimensions and wings in three dimensions. Contrary to existing neural networks that are often designed to predict aerodynamic quantities of interest, the proposed network considers as output the pressure at a number of selected points on the aerodynamic shape. The proposed multi-output neural network is compared with other approaches found in the literature. Furthermore, a detailed comparison of the proposed neural network with the popular proper orthogonal decomposition method is presented. The numerical results, involving high dimensional problems with flow and geometric parameters, show the benefits of the proposed approach.

Keywords: neural network, proper orthogonal decomposition, CFD, NURBS, aerofoil, wing.

1. Introduction

Computational fluid dynamics (CFD) is heavily used to approximate the aerodynamic forces on aerofoils in different engineering applications [1, 2, 3]. During the design and optimisation stages of aerodynamic components, the simulations to be performed involve a large number of parameters related to

*Corresponding author
Email address: o.hassan@swansea.ac.uk (Oubay Hassan)

the geometry and flow conditions. In this scenario, the simulation of all possible configurations is not affordable. In addition, it is worth noting that the parameters involved in the simulations often contain a certain level of uncertainty [4, 5].

The use of reduced order models (ROM) has become a popular alternative to alleviate the computational burden associated to CFD simulations involving a large number of parameters [6]. Some of the more popular ROMs are the reduced basis method [7], the proper orthogonal decomposition (POD) [8, 9, 10, 11, 12], the non-linear kernel principal component analysis (kPCA) [13, 14] and the proper generalised decomposition (PGD) [15, 16, 17]. Although non-intrusive approaches exist [18, 19, 20], the applicability of the PGD is often limited by the number of parameters to be considered.

Artificial neural networks (NN) have also been successfully employed as a ROM in many areas of science and engineering, including CFD [21, 22]. One of the first applications of NNs as a data-driven approach for the design of aerofoils can be found in [23]. One of the first applications of NN to study geometrically parametrised aerofoils was presented in the 1990s [24]. Physics-based NNs have also become another popular option in the field of aerodynamic modelling. The authors in [25] employ supervised NNs that respect any given laws of physics described by generalised non-linear partial differential equations. Researchers have also integrated the physical laws of the Navier-Stokes equations in NNs [26, 27].

Physics-inspired NN models have also been built in which NN is used to map the explanatory variables to the POD coefficients such that for a new set of inputs, the POD coefficients are obtained in the lower dimensional space of the reduced order model [28, 29, 30]. The kPCA, a non-linear projection method to a higher dimensional space, has mainly been used as a feature extraction method in the explanatory variables and an alternative regression method is used to map these feature variables, expressed in higher dimensions, to the response variables [14, 31]. The reconstruction from its principal components to the feature space is always possible, however the authors in [32, 33] underlines the difficulties in the reconstruction to the input space.

In the last two decades, the use of NNs in CFD applications has grown significantly. In [34], the authors use NNs to predict the pressure over a geometrically parametrised wing. The use of the so-called PARSEC aerofoil representation [35] has also been extensive employed to train NNs for the prediction of aerodynamic coefficients [36, 37, 38]. In [39], an extension to hybrid geometric parametrisations using the PARSEC representation and Bezier curves is presented. More recently, approaches based on aerofoil images, rather than a mathematical description, have also been considered [40].

One of the most attractive properties of ROMs and NNs is that, once the ROM or NN is built, predictions can be performed in almost real time. This enables the use of such models for the fast evaluations of an objective function in optimisation and inverse problems. The use of NNs to perform directly the optimisation has also been considered in the literature [41, 39, 42].

In this work, a multi-output NN is proposed to predict the aerodynamic coefficients of aerofoils and wings using inviscid compressible CFD data. The output consists of the pressure evaluated at a number of points on the aerofoil surface. Here this is done using a single NN, rather than as many NNs as the number of points on the surface, as performed in [34]. The performance of the proposed NN is compared with the most common approach found in the literature, where a NN with a single output, corresponding to the lift, is considered [43, 36, 44, 37, 24, 40]. The proposed approach is also compared to the POD in the numerical examples in two dimensions.

The numerical examples presented consider parameters corresponding to inflow conditions and geometries. For problems with parametrised inflow conditions, the selected range for the angle of attack and the Mach number leads to subsonic and transonic flow regimes. This situation is known to be challenging as some ROMs have shown difficulties in accurately predicting the flow features across different regimes. For problems involving parametrised geometries, non-uniform rational B-Splines [45] are considered as the geometric representation. The control points of the NURBS curves or surfaces are considered to be the parameters. Detailed numerical studies are also presented to select the optimal hyperparameters of the various configurations of NN employed and the numerical parameters of the POD.

The remainder of the paper is structured as follows. Section 2 briefly recalls the Euler equations for an inviscid compressible flow and the finite volume scheme used here to obtain the CFD data. In sections 3 and 4 the fundamentals of NNs and the POD are presented, respectively. In section 5 three numerical examples in two dimensions are presented. The first example considers a problem where the angle of attack and the Mach number are the parameters. The proposed multi-output NN is built and compared against traditional NNs predicting the aerodynamic coefficients. Numerical studies are also presented to show the influence of the NN hyperparameters on the accuracy of the predictions. The second example compares the proposed NN to the POD for a problem where the angle of attack and the Mach number are the parameters. The last example of section 5 compares the proposed NN and the POD for a problem with 25 geometric parameters corresponding to the position of the NURBS control points defining the aerofoil. Both subsonic and transonic cases are analysed. In section 6, three examples in three dimensions are also presented. The proposed multi-output NN is compared against two other NNs predicting the aerodynamic coefficients directly, the single output NN and the three outputs NN. The first example considers a problem where the angle of attack and Mach number are varied and shows the influence of the accuracy of the CFD data on the accuracy of the NN predictions. The second example involves the NN predictions on a geometrically parametrised wing with 50 geometric parameters. The last example of section 6 uses NN to perform predictions on a deforming wing, in which the Mach number, angle of attack, geometric twist in the spanwise direction and wingtip deflection in the z-plane are varied.

2. Finite volume solution of inviscid compressible flows

2.1. Governing equations

The fluid flow problems considered in this work are governed by the Euler equations for an inviscid compressible fluid. The strong form of the problem, in a computational domain $\Omega \subset \mathbb{R}^d$ and in the absence of external volume forces, can be written as

$$\begin{aligned} \boldsymbol{U}_t + \boldsymbol{\nabla} \cdot \boldsymbol{F}(\boldsymbol{U}) &= \boldsymbol{0} \quad \text{in } \Omega \times (\,0, T] \\ \boldsymbol{U} &= \boldsymbol{U}_0 \quad \text{in } \Omega \times \{0\} \\ \boldsymbol{B}(\boldsymbol{U}, \boldsymbol{U}^\infty) &= \boldsymbol{0} \quad \text{in } \partial\Omega \times (\,0, T]. \end{aligned} \tag{1}$$

The vector of conservation variables, $\boldsymbol{U}$, and the flux tensor, $\boldsymbol{F}$, are given by

$$\boldsymbol{U} := \begin{Bmatrix} \rho \\ \rho \boldsymbol{v} \\ \rho E \end{Bmatrix}, \qquad \boldsymbol{F} := \begin{bmatrix} \rho \boldsymbol{v}^T \\ \rho \boldsymbol{v} \otimes \boldsymbol{v} + p\mathbf{I}_d \\ (\rho E + p)\boldsymbol{v}^T \end{bmatrix}, \tag{2}$$

$\boldsymbol{U}_0$ denotes the initial condition, T is the final time and $\boldsymbol{B}$ is the generic flux used to define the boundary conditions. In the above expressions ρ is the density, $\rho\boldsymbol{v}$ is the momentum, $\rho\boldsymbol{E}$ is the total energy per unit volume, p is the pressure and $\mathbf{I}_d$ is the identity matrix of dimension d.

The Euler equations are closed with the equation of state for a perfect polytropic gas, in the form

$$p = (\gamma - 1)\rho \left(E - \frac{1}{2} \|\boldsymbol{v}\|^2 \right), \tag{3}$$

where $\gamma = 1.4$, is the ratio of the specific heats for air.

2.2. Vertex-centred finite volume method

Vertex-centred finite volume (FV) methods form the basis of many industrial and research codes and have proved to be competitive when simulating steady flows [46, 47].

In the FV context, a control volume Ω_i, associated to a node $\boldsymbol{x}_i$, is formed by joining the edge midpoints and the element centroids of the edges connected to the node $\boldsymbol{x}_i$ and the elements sharing $\boldsymbol{x}_i$ respectively.

The spatial discretisation of the vertex-centred FV method employs a constant approximation of the solution in each control volume, Ω_i. Second-order convergence is achieved by re-constructing a linear approximation to the fluxes on each edge of the control volume. More information about the solver utilised in this work to produce the CFD data can be found in [48].

2.3. Computation of the aerodynamic coefficients

The calculations of the lift, drag and moment coefficients for a given configuration are written as

$$C_L = \frac{1}{q_\infty S_\Gamma} \int_\Gamma p(\boldsymbol{n}^w \cdot \boldsymbol{n}^\infty) d\Gamma, \tag{4}$$

$$C_D = \frac{1}{q_\infty S_\Gamma} \int_\Gamma p(\boldsymbol{n}^w \cdot \boldsymbol{t}^\infty) d\Gamma, \tag{5}$$

$$\boldsymbol{C_M} = \frac{1}{q_\infty S_\Gamma l} \int_\Gamma p(\boldsymbol{r} \times \boldsymbol{n}^w) d\Gamma, \tag{6}$$

where Γ is the integration surface, S_Γ is the reference area, taken to be the chord length in two dimensions or the planform area in three dimensions, q_∞ is the free-stream dynamic pressure, p is the static pressure, $\boldsymbol{n}^w$ denotes the outward unit normal vector to the wall surface, $\boldsymbol{t}^\infty$ and $\boldsymbol{n}^\infty$ are the tangential and normal unit vectors with respect to the direction of the free-stream velocity. The vector obtained in equation 6 represents the moments in the x, y and z directions. In this work, only the pitching moment is reported, corresponding to the moment about the y-axis in three dimensions. A length scale, l, is introduced and this is usually taken to be the chord length of the aerofoil in two dimensions or the chord length of mid-span aerofoil cross section of the wing in three dimensions. The vector $\boldsymbol{r}$ denotes the position of a point, $\boldsymbol{x}$, on Γ with respect to a reference point $\boldsymbol{x}_{\texttt{ref}}$, namely $\boldsymbol{r} = \boldsymbol{x} - \boldsymbol{x}_{\texttt{ref}}$. The reference point is taken as the aerodynamic centre in two dimensions or the centre of gravity in three dimensions, unless stated otherwise.

3. Artificial neural networks

Artificial NNs are an arrangement of neurons organised by layers. In the networks considered here, the neurons of each layer are connected to the neurons of the previous and subsequent layers. Each neuron has an associated value and each connection has an associated weight. This particular case is also referred to as multi-layer perceptron, which is a class of feed-forward NN. The first and the last layers correspond to the inputs and outputs, and the remaining layers, called hidden layers, are numbered from $l = 1$ to $l = \mathtt{n_L}$, with $\mathtt{n_L}$ being the number of hidden layers [49].

During the so-called forward propagation, the value associated to each neuron is computed by using the values associated to the connected neurons in the previous layer, the weights of the connections and an activation function F^l. More precisely, the value of the j-th neuron in the layer $l+1$, denoted by z_j^{l+1}, is computed as

$$z_j^{l+1} = F^l \left(\sum_{i=1}^{\mathtt{n}_\mathtt{N}^l} \theta_{ij}^l z_i^l + b_j^l \right), \tag{7}$$

where b_j^l is a bias that is introduced to enhance the approximation properties of the network, θ_{ij}^l denotes the weight of the connection between the i-th neuron of the layer l and the j-th neuron of the layer $l+1$ and $\mathtt{n}_\mathtt{N}^l$ is the number of neurons in the layer l. An illustration of a generic multi-layer perceptron NN is shown in figure 1.

A training case is defined by a vector of N inputs, $\boldsymbol{x} = \{x_1, \dots, x_N\}^T$, and a vector of M outputs, $\boldsymbol{y}(\boldsymbol{x}) = \{y_1(\boldsymbol{x}), \dots, y_M(\boldsymbol{x})\}^T$. Given a set of $\mathtt{n_{Tr}}$ training cases, $\boldsymbol{x}^k = \{x_1^k, \dots, x_N^k\}$ and $\boldsymbol{y}^k = \{y_1^k, \dots, y_M^k\}$, for $k = 1, \dots, \mathtt{n_{Tr}}$, the cost function is defined as

$$C(\boldsymbol{\theta}) = \frac{1}{M \mathtt{n_{Tr}}} \sum_{k=1}^{\mathtt{n_{Tr}}} \sum_{i=1}^{\mathtt{n}_\mathtt{N}^{\mathtt{n_L}+1}} \left[y_i^k(\boldsymbol{x}^k) - h_i^k(\boldsymbol{\theta}) \right]^2 + \frac{\lambda}{2\mathtt{n_{Tr}}} \sum_{l=0}^{\mathtt{n_L}} \sum_{i=1}^{\mathtt{n}_\mathtt{N}^{l+1}} \sum_{j=1}^{\mathtt{n}_\mathtt{N}^l} (\theta_{ij}^l)^2. \tag{8}$$

The values h_i^k correspond to the predicted outputs, computed using a forward propagation, starting from the input values $z_i^0 = x_i^k$ for $k = 1, \dots, \mathtt{n_{Tr}}$ and

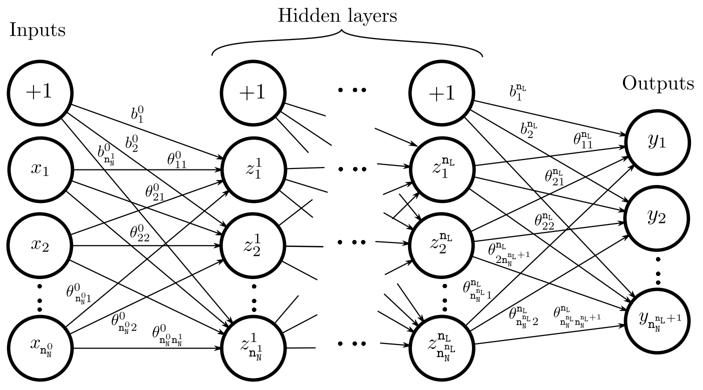

Inputs
Hidden layers
Outputs
$+1$
x_1
x_2
$x_{\mathtt{n}_{\mathtt{N}}^0}$
z_1^1
z_2^1
$z_{\mathtt{n}_{\mathtt{N}}^1}^1$
$z_1^{\mathtt{n}_{\mathtt{L}}}$
$z_2^{\mathtt{n}_{\mathtt{L}}}$
$z_{\mathtt{n}_{\mathtt{N}}^{\mathtt{n}_{\mathtt{L}}}}^{\mathtt{n}_{\mathtt{L}}}$
y_1
y_2
$y_{\mathtt{n}_{\mathtt{N}}^{\mathtt{n}_{\mathtt{L}}+1}}$
b_1^0
b_2^0
$b_{\mathtt{n}_{\mathtt{N}}^1}^0$
θ_{11}^0
θ_{21}^0
θ_{22}^0
$\theta_{\mathtt{n}_{\mathtt{N}}^0 1}^0$
$\theta_{\mathtt{n}_{\mathtt{N}}^0 2}^0$
$\theta_{\mathtt{n}_{\mathtt{N}}^0 \mathtt{n}_{\mathtt{N}}^1}^0$
$b_1^{\mathtt{n}_{\mathtt{L}}}$
$b_2^{\mathtt{n}_{\mathtt{L}}}$
$\theta_{11}^{\mathtt{n}_{\mathtt{L}}}$
$\theta_{21}^{\mathtt{n}_{\mathtt{L}}}$
$\theta_{22}^{\mathtt{n}_{\mathtt{L}}}$
$\theta_{2\mathtt{n}_{\mathtt{N}}^{\mathtt{n}_{\mathtt{L}}+1}}^{\mathtt{n}_{\mathtt{L}}}$
$\theta_{\mathtt{n}_{\mathtt{N}}^{\mathtt{n}_{\mathtt{L}}} 1}^{\mathtt{n}_{\mathtt{L}}}$
$\theta_{\mathtt{n}_{\mathtt{N}}^{\mathtt{n}_{\mathtt{L}}} 2}^{\mathtt{n}_{\mathtt{L}}}$
$\theta_{\mathtt{n}_{\mathtt{N}}^{\mathtt{n}_{\mathtt{L}}} \mathtt{n}_{\mathtt{N}}^{\mathtt{n}_{\mathtt{L}}+1}}^{\mathtt{n}_{\mathtt{L}}}$

Figure 1: Schematic representation of a multi-layer perceptron NN.

$i = 1, \dots, \mathtt{n}_{\mathtt{N}}^0$. The number of neurons in the input layer is taken as $\mathtt{n}_{\mathtt{N}}^0 = N$ and, similarly, the number of neurons in the output layer is taken as $\mathtt{n}_{\mathtt{N}}^{\mathtt{n}_{\mathtt{L}}+1} = M$.

The first term of equation (8) is the mean square error and it measures the discrepancy between the outputs and the NN predictions. The second term of equation (8) is a penalty term and it corresponds to one of the most popular techniques used in NNs to avoid overfitting [50, 51]. The bias is not employed within the second term as it has been shown that this usually has a negative effect on the NN approximation properties [52].

The goal of the so-called training stage is to obtain the weights associated to all the connections of the NN that minimise the cost function of equation (8). In this work a backpropagation gradient descent search algorithm is employed [53]. A low pass filter is applied to alleviate the common difficulty of gradient based methods to reach a global minimum in problems where multiple local minimums are present. More precisely, the momentum gradient descent approach is employed, where the iterative process to compute the optimum weights at the iteration $r + 1$ is given by

$$\theta_{ij}^{l,r+1} = (1+\eta)\theta_{ij}^{l,r} - \eta\theta_{ij}^{l,r-1} - \tau(1-\eta)\frac{\partial C}{\partial \theta_{ij}^{l,r}}, \tag{9}$$

where $\eta \in [0, 1)$ is the momentum coefficient of the first order smoothing filter and τ is the step length to move in the direction of the gradient.

Although other minimisation algorithms have proven advantageous in some situations [54], the backpropagation approach employed here has demonstrated a good performance for training NNs of moderate size [55].

The design of the NN implemented in this work requires the choice of an activation function in each layer, the number of hidden layers, the number of neurons in each hidden layer and the overfitting parameter λ. Among all the possible options, two classes of functions are considered, namely a sigmoid function and a linear function, given by

$$S(x) = \frac{1}{1+e^{-x}} \qquad \text{and} \qquad L(x) = x, \tag{10}$$

respectively. When the NN is designed as a function approximant, the activation functions are selected as $F^l = S$ for $l = 0, \dots \mathtt{n_L}-1$ and $F^{\mathtt{n_L}} = L$. In contrast, for NN designed as classifiers, all the activation functions are taken as the sigmoid, $F^l = S$ for $l = 0, \dots \mathtt{n_L}$.

The cost function of equation (8) is used when the NN is designed as a function approximant, whereas for NNs designed as classifiers, the cost function is taken as

$$C(\boldsymbol{\theta}) = -\frac{1}{M\mathtt{n_{Tr}}} \sum_{k=1}^{\mathtt{n_{Tr}}} \sum_{i=1}^{\mathtt{n_N^{n_L+1}}} \left\{ y_i^k(\boldsymbol{x}^k) \log\left[h_i^k(\boldsymbol{\theta})\right] + [1-y_i^k(\boldsymbol{x}^k)] \log\left[1-h_i^k(\boldsymbol{\theta})\right] \right\} \\ + \frac{\lambda}{2\mathtt{n_{Tr}}} \sum_{l=0}^{\mathtt{n_L}} \sum_{i=1}^{\mathtt{n_N^{\mathit{l}+1}}} \sum_{j=1}^{\mathtt{n_N^{\mathit{l}}}} (\theta_{ij}^l)^2. \tag{11}$$

The remaining parameters involved in the network design are known to be problem dependent. The numerical examples in sections 5 and 6 will show the influence of these parameters on the approximation properties of the NNs for the problems under consideration.

The algorithm to compute the optimum weights of the NN also requires the choice of the parameters η and τ as well as an initial guess for the iterative process given by equation (9). This choice is usually based on numerical experiments and some recommendations can be found in the literature [49]. Following [56], the initialisation of the weights for a layer l uses a uniform distribution within the interval $[-\sqrt{a^l}, \sqrt{a^l}]$, with $a^l = 6/(\mathtt{n}_\mathtt{N}^l + \mathtt{n}_\mathtt{N}^{l+1})$.

4. Proper orthogonal decomposition

Proper orthogonal decomposition (POD) is a popular model order reduction technique that has been successfully applied in many areas of science and engineering. The most common applications in CFD are found in transient fluid flow problems where the POD produces surrogate models [8, 9, 11, 10]. The application of POD to CFD parametric problems, as considered here, is more recent and some examples can be found in [57, 58, 12].

Let us consider a set of $\mathtt{n}_\mathtt{Tr}$ training cases, or snapshots, defined by a set of inputs and outputs, $\boldsymbol{x}^k = \{x_1^k, \dots, x_N^k\}$ and $\boldsymbol{y}^k(\boldsymbol{x}^k) = \{y_1^k(\boldsymbol{x}^k), \dots, y_M^k(\boldsymbol{x}^k)\}$, respectively, for $k = 1, \dots, \mathtt{n}_\mathtt{Tr}$. The matrix of snapshots is formed with all the available outputs as

$$\mathbf{Y}(\boldsymbol{x}^1, \boldsymbol{x}^2, \dots, \boldsymbol{x}^{\mathtt{n}_\mathtt{Tr}}) = \begin{bmatrix} y_1^1(\boldsymbol{x}^1) & y_1^2(\boldsymbol{x}^2) & \cdots & y_1^{\mathtt{n}_\mathtt{Tr}}(\boldsymbol{x}^{\mathtt{n}_\mathtt{Tr}}) \\ y_2^1(\boldsymbol{x}^1) & y_2^2(\boldsymbol{x}^2) & \cdots & y_2^{\mathtt{n}_\mathtt{Tr}}(\boldsymbol{x}^{\mathtt{n}_\mathtt{Tr}}) \\ \vdots & \vdots & & \vdots \\ y_M^1(\boldsymbol{x}^1) & y_M^2(\boldsymbol{x}^2) & \cdots & y_M^{\mathtt{n}_\mathtt{Tr}}(\boldsymbol{x}^{\mathtt{n}_\mathtt{Tr}}) \end{bmatrix}. \tag{12}$$

The singular value decomposition of the snapshot matrix is given by

$$\mathbf{Y} = \mathbf{V}\boldsymbol{\Sigma}\mathbf{W}^*, \tag{13}$$

where $\mathbf{V}$ is a unitary matrix of dimension $M \times M$, whose columns, $\mathbf{V}_j$ for $j = 1, \dots, M$, are the left singular vectors of $\mathbf{Y}$, $\boldsymbol{\Sigma}$ is a rectangular diagonal matrix of dimension $M \times \mathtt{n}_\mathtt{Tr}$ whose entries, α_j for $j = 1, \dots, \min\{M, \mathtt{n}_\mathtt{Tr}\}$, are the singular values of $\mathbf{Y}$, $\mathbf{W}$ is a unitary matrix of dimension $\mathtt{n}_\mathtt{Tr} \times \mathtt{n}_\mathtt{Tr}$, whose columns, $\mathbf{W}_j$ for $j = 1, \dots, \mathtt{n}_\mathtt{Tr}$, are the right singular vectors of $\mathbf{Y}$ and the superscript $*$ denotes the conjugate transpose.

Since the columns of $\mathbf{V}$ form an orthonormal basis, each column of the snapshot matrix $\mathbf{Y}$, corresponding to a set of inputs $\boldsymbol{x}^k$, for $k = 1, \dots, \mathtt{n}_\mathtt{Tr}$, can be reconstructed using

$$\boldsymbol{y}^k(\boldsymbol{x}^k) = \sum_{j=1}^{M} \alpha_j^k(\boldsymbol{x}^k)\mathbf{V}_j, \tag{14}$$

where α_j^k are the POD coefficients and the vectors $\mathbf{V}_j$ are referred to as POD modes.

After the POD coefficients and modes are computed, the outputs, $\boldsymbol{y}$, for a given set of inputs, $\boldsymbol{x}$, not present in the set of training examples, can be approximated by

$$\boldsymbol{y}(\boldsymbol{x}) \approx \sum_{j=1}^{M} \alpha_j(\boldsymbol{x}) \mathbf{V}_j, \tag{15}$$

where the coefficients α_j are obtained by interpolating the POD coefficients. In practice, the number of modes selected to construct the approximation $\boldsymbol{y}(\boldsymbol{x})$ is lower than M. However, in the examples shown in this work all the POD modes are considered to ensure that the accuracy of the POD approximation is not influences by the number of modes considered.

Following the work in [11], a radial basis function (RBF) interpolation is considered here, which is suitable for problems involving a large number of inputs. The interpolation is given by

$$\alpha_j(\boldsymbol{x}) = \sum_{k=1}^{\mathtt{n_{Tr}}} w_j^k \Theta(\|\boldsymbol{x} - \boldsymbol{x}^k\|), \tag{16}$$

for $j = 1, \dots, M$, where w_j^k, for $k = 1, \dots, \mathtt{n_{Tr}}$, is the set of unknown coefficients and Θ is a radial basis function. The multi-quadric function

$$\Theta(r) = \sqrt{r^2 + \varepsilon^2} \tag{17}$$

is considered, where ε is a numerical parameter that dictates the radius of influence of the data.

The coefficients w_j^k in equation (16) are computed by solving a set of independent linear system of equations,

$$\mathbf{A}\mathbf{w}^j = \mathbf{f}^j \tag{18}$$

for $j = 1, \dots, M$, with $A_{IJ} = \Theta(\|\boldsymbol{x}^I - \boldsymbol{x}^J\|)$ and $f_I^j = \alpha_j^I(\boldsymbol{x}^I)$.

5. Numerical examples in two dimensions

5.1. Aerodynamic predictions on a NACA0012 aerofoil at various inflow conditions

The first example considers the computation of the aerodynamic coefficients of a NACA0012 aerofoil at a free stream Mach number, M_∞, and an angle of attack, α, in predefined intervals $I_M = (0.3, 0.9)$ and $I_\alpha = (-5^\circ, 11^\circ)$, respectively. This range of the inflow conditions leads to both subsonic and transonic flows.

Three NNs are considered and compared. The first network considers M_∞ and α as inputs and, the lift, drag or moment coefficient as a single output. The second NN also considers M_∞ and α as inputs and has three outputs, corresponding to the three aerodynamic coefficients. The third network considers M_∞ and α as inputs and the output is the pressure at a user defined set of points on the aerofoil. The set of points considered corresponds to the 300 mesh nodes used to discretise the aerofoil. The first and second networks has previously been considered [43, 36, 44, 37, 24, 40], whereas the third network, to the authors' best knowledge, has not been investigated previously.

All three networks are trained using a dataset of $\mathtt{n}_{\mathtt{Tr}} = 40$ simulations performed with the vertex-centred finite volume method [48]. The training set is selected by using the latin hypercube sampling [59] in $I_M \times I_\alpha$. The test set consists of $\mathtt{n}_{\mathtt{Te}} = 119$ cases in $I_M \times I_\alpha$ and employs an equally-spaced distribution in $\bar{I}_M$ with step 0.1. Similarly, an equally-spaced distribution in I_α with step 1° is considered for each value of M_∞. Figure 2 shows the design sampling space used to define the training and test data. It can be observed that the test set contains a number of points that will induce an extrapolation, i.e. they are outside of the convex hull defined by the training set. Therefore, this example is also useful to evaluate the performance of the different networks when the predictions involve extrapolation. The learning rate and the momentum coefficient of the networks considered in this work, are taken as $\tau \in [0.005, 0.5]$ and $\eta = 0.9$ respectively. These two parameters have been obtained after performing experiments with

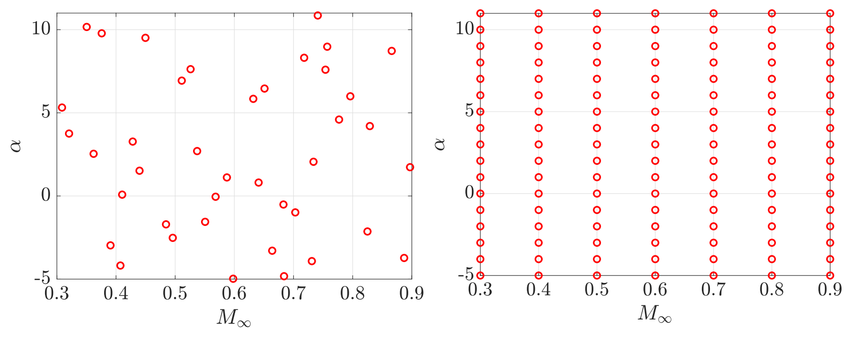


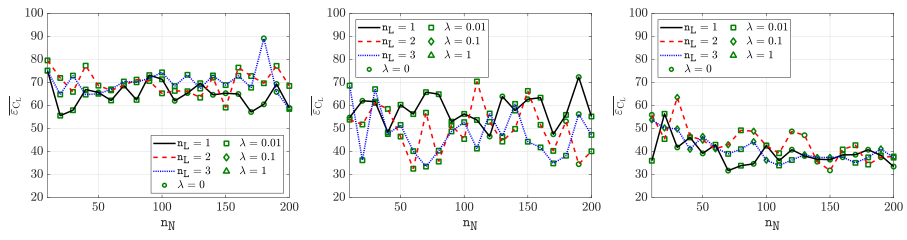
Figure 2: The sampling space used to define the training and test dataset using $\texttt{n}_{\texttt{Tr}} = 40$ and $\texttt{n}_{\texttt{Te}} = 119$ cases respectively.

the two networks considered, and are similar to the parameters used in other studies [49].

The first numerical experiment explores the accuracy of the predicted aerodynamic coefficients as a function of the number of hidden neurons, $\texttt{n}_{\texttt{N}}$, and the number of hidden layers, $\texttt{n}_{\texttt{L}}$, for the three networks considered and for different values of the over-fitting parameter, λ. The cost function is implemented with the regularisation term to avoid the need to divide the dataset into validation and training cases. This choice implies that the over-fitting parameter has to be tuned, but it ensures that the whole set of available data can be used for training. The error in the lift prediction of the k-th test case, measured in lift counts, is defined as

$$\varepsilon_{C_L,k} = |C_L(k) - C_L^\star(k)| \times 10^3, \tag{19}$$

where C_L is the lift coefficient from the CFD solver and $C_L^\star$ is the lift coefficient predicted by the ROM or NN. In the case when the drag or moment coefficient is predicted, the error measured in drag counts or moment counts is calculated in the ten thousandths decimal. It should also be noted that the third network outputs the pressure on the aerofoil and the computation of the lift is performed after the pressure distribution is predicted. In the numerical examples, two error measures are considered to assess the overall performance of the neural networks,



(a) Network 1 (b) Network 2 (c) Network 3

Figure 3: Mean value of the error measured in lift counts, $\overline{\varepsilon_{C_L}}$, as a function of the number of hidden neurons, $\mathtt{n_N}$, and the number of hidden layers, $\mathtt{n_L}$.

namely the mean value of the error measured in the test set, defined as

$$\overline{\varepsilon_{C_L}} = \frac{1}{\mathtt{n_{Te}}} \sum_{k=1}^{\mathtt{n_{Te}}} \varepsilon_{C_L,k}, \tag{20}$$

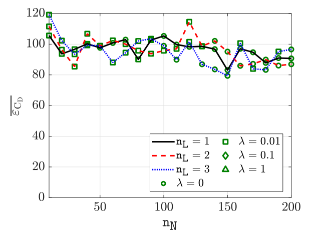
and the maximum value of the error measured in the test set, defined as

$$\varepsilon_{C_L,\mathtt{max}} = \max_{k=1,\dots,\mathtt{n_{Te}}} \{\varepsilon_{C_L,k}\}. \tag{21}$$

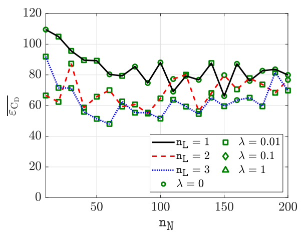
Figure 3 shows the mean value of the error, measured in lift counts, as a function of the number of hidden neurons, $\mathtt{n_N}$, and the number of hidden layers, $\mathtt{n_L}$. The results in figure 3(a) show that for the first network, with a single output corresponding to the lift, the highest accuracy is obtained with one hidden layer and a large number of neurons, namely $\mathtt{n_N} \in [20, 30]$ and $\mathtt{n_N} \in [170, 200]$. The best accuracy obtained with this network is $\overline{\varepsilon_{C_L}} = 55$. Similar accuracy is also provided by using three hidden layers and 200 hidden neurons. However, the results show that a small variation in the number of neurons can lead to a substantial increase in the error, suggesting a lack of robustness.

The second network, with three aerodynamic coefficients as outputs, provide more accurate results with two or three hidden layers over the range of number of neurons covered, $\mathtt{n_N} = [60, 70, 170, 190]$, however, higher deviations from the mean error are observed in the first network. This indicates that this network has a high dependence on the number of hidden layers and neurons.

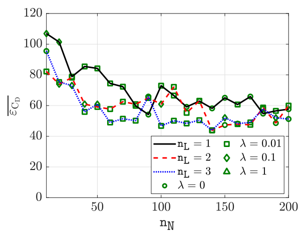
The results for the third network, with multiple outputs corresponding to the pressure on the aerofoil, are shown in figure 3(c). It can be observed that



(a) Network 1



(b) Network 2



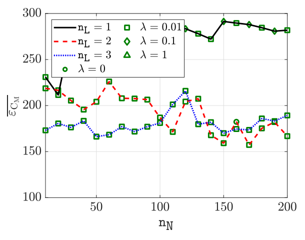
(c) Network 3

Figure 4: Mean value of the error measured in drag counts, $\overline{\varepsilon_{C_D}}$, as a function of the number of hidden neurons, $\mathtt{n}_\mathtt{N}$, and the number of hidden layers, $\mathtt{n}_\mathtt{L}$.

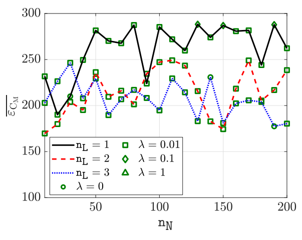
the accuracy is less dependent on the number of neurons in the hidden layers and the mean error has a tendency to decrease as the number of neurons is increased. The best accuracy obtained with this network is $\overline{\varepsilon_{C_L}} = 31$, when one hidden layer with 70 hidden neurons is used. There are no significant differences in the accuracy provided by the network with one hidden layer and a number of neurons $\mathtt{n}_\mathtt{N} \in [120, 180]$ or with three hidden layer and a number of neurons in a similar range. This illustrates its robustness when compared to the first two networks. For the third network, there is no advantage in using more than one hidden layer. This shows that the level of non-linearity required to model the pressure on the aerofoil is lower than the level of non-linearity required to directly model the lift as an output.

Figure 4 shows the mean value of the error, measured in drag counts, as a function of the number of hidden neurons, $\mathtt{n}_\mathtt{N}$, and the number of hidden layers, $\mathtt{n}_\mathtt{L}$ for the three different NNs. The first NN provides less accurate results, whereas the proposed NN provides the most accurate results and is less sensitive to the choice of the hyperparameters. The second NN is also capable of providing accurate results if only two and three hidden layers are used.

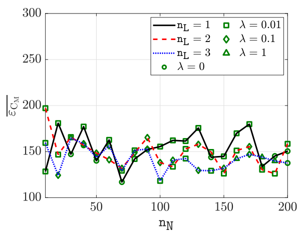
Finally, figure 5 shows the mean value of the error, measured in moment counts, as a function of the number of hidden neurons, $\mathtt{n}_\mathtt{N}$, and the number of hidden layers, $\mathtt{n}_\mathtt{L}$ for the three different NNs. For this aerodynamic quantity of interest, the superiority of the proposed NN is clearly observed. The first and



(a) Network 1

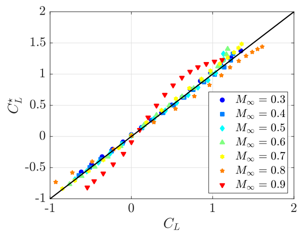


(b) Network 2

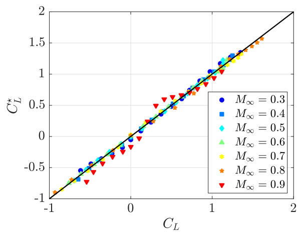


(c) Network 3

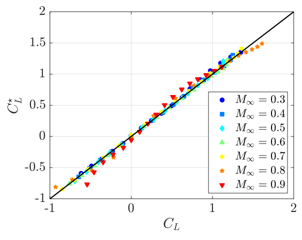
Figure 5: Mean value of the error measured in moment counts, $\overline{\varepsilon_{C_M}}$, as a function of the number of hidden neurons, $\mathtt{n_N}$, and the number of hidden layers, $\mathtt{n_L}$.



(a) Network 1



(b) Network 2



(c) Network 3

Figure 6: Regression plot for the lift coefficient as a function of the free-stream Mach number M_∞.

second NNs are not able to provide a single case where the prediction provides an error below 150 counts, whereas the proposed NN is consistently providing much lower errors. For this quantity of interest it seems advantageous to employ more than one hidden layer.

As the numerical experiments performed on the predictions of the drag and moment produced similar deductions to the predictions of the lift, only the lift quantity will be presented in the remaining examples in two dimensions.

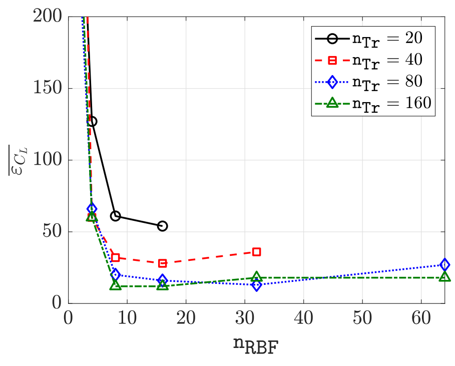
To illustrate the performance of the networks in the different flow regimes considered, figure 6 shows the regression plots for the lift coefficient using the three types of network employed. Different symbols are used for different free-stream Mach numbers to show the accuracy of the predictions in terms of the flow regime. The results shown correspond to the best accuracy measured in the mean value of the error in lift counts obtained with the networks considered. The first network employs one hidden layer and 20 hidden neurons, the second network has two hidden layers and 60 hidden neurons, and finally, the third network is selected to have one hidden layer and 70 hidden neurons. The results for the first and second networks, reported in figures 6(a), and 6(b), show a large deviation for $M_\infty = 0.8$ and $M_\infty = 0.9$. The higher accuracy of the third network is clearly observed in figure 3(c), with a much lower deviation in all cases. The results also show that the only sizeable deviation in the predictions with the proposed NN are observed for $M_\infty = 0.9$. This corresponds to cases which contains strong shocks and where an extrapolation is performed, as seen in figures 2.

5.2. Lift prediction on an RAE2822 aerofoil at various inflow conditions

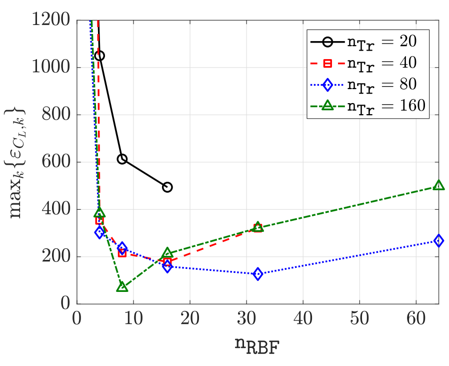
The second example considers the prediction of the lift on the RAE2822 aerofoil at a free stream Mach number, M_∞, and an angle of attack, α, in predefined intervals $I_M = (0.3, 0.9)$ and $I_\alpha = (-5^\circ, 12^\circ)$ respectively. The flow in this range will be either subsonic or transonic.

This example compares the performance of the multi-output NN against a traditional reduced order modelling technique used in aerodynamic modelling, the POD [60]. The influence of the number of training cases considered on the accuracy of the predictions is also studied by considering training sets ranging from $\mathtt{n_{Tr}} = 20$ up to $\mathtt{n_{Tr}} = 160$ cases.

The performance of the POD method with RBFs used to perform the interpolation [11], is evaluated. In all cases, POD modes are computed from the training cases, corresponding to the pressure over the 300 points describing the aerofoil, rather than estimating the value of the lift directly. To predict the value of the pressure for a given test case, the RBF interpolation requires the selection of the radius of influence, that is the parameter ε in the multi-quadric function of equation (17). To study the effect of this numerical parameter, figure 7 shows the evolution of the mean and maximum error, measured in lift counts, as a function of the number of cases that are included in the interpola-



(a) Mean error



(b) Maximumn error

Figure 7: Evolution of the mean and maximum error on the test set, measured in lift counts, as a function of the number of cases used to perform the interpolation with RBFs, $\mathtt{n}_{\mathtt{RBF}}$.

tion, $\mathtt{n}_{\mathtt{RBF}}$, for an increased value of the radius of influence. The results show that the optimal value of the radius, when the number of snapshots is large enough, i.e. $\mathtt{n}_{\mathtt{Tr}} = 160$, is such that the RBFs use the values of the closest eight cases to perform the interpolation. Larger values of the radius do not significantly affect the mean value of the error, as shown in figure 7(a), but lead to a loss of accuracy when the maximum error is considered, as shown in figure 7(b).

Next, the best accuracy of the proposed NN and the POD with RBFs is compared in the test dataset when using dataset of size ranging from $\mathtt{n}_{\mathtt{Tr}} = 20$ to $\mathtt{n}_{\mathtt{Tr}} = 160$ cases in the training. The NN employed has one hidden layer and 98 neurons and the over-fitting parameter is set to $\lambda = 0$. For the POD, eight cases are considered to perform the interpolation using the RBFs. In both cases, the prediction of the pressure is performed at the 300 points used to discretise the aerofoil and the lift coefficient is computed after the chosen reduced order model is employed.

Figure 8 compares the performance of the reduced order models considered, the NN and POD, by reporting the mean and maximum error as a function of the number of training cases. The results show that NN provides the best accuracy in all cases when the accuracy is measured as the mean error. The

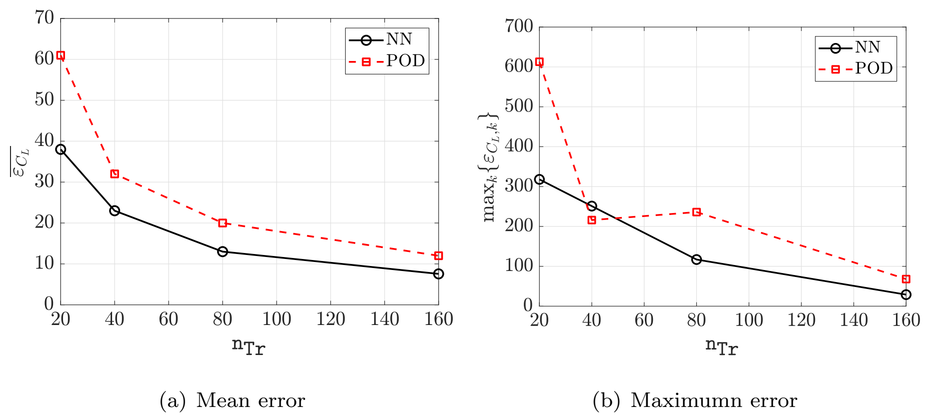


Figure 8: Evolution of the mean and maximum error on the test set, measured in lift counts, as a function of the number of training cases, or snapshots, for the two reduced order models employed.

trained NN model can provide a mean error of seven lift counts when using $\mathtt{n_{Tr}} = 160$ training examples. In contrast with the NN, POD follows a similar variation of the mean error as the number of training examples is increased, with a higher mean error of 12 lift counts, measured in the test set, for the same number of training data. Figure 8(b) compares the maximum error measured in lift counts and it shows that for almost any number of training samples, the maximum error of the POD is more than twice that of the NN, except when using $\mathtt{n_{Tr}} = 40$ cases where the results are comparable. NN has a maximum error of 29 lift counts and POD has 68 lift counts when using $\mathtt{n_{Tr}} = 160$ cases in the training.

Figure 9 compares the accuracy of the proposed NN and the POD. The histogram represents the relative frequency of the error, measured in lift counts, for all the test cases. The results show that both approaches are able to provide an error below 10 lift counts for almost 70% of the test cases, with the NN achieving a marginally higher percentage. The first significant difference is that the NN provides a prediction with an error below 20 lift counts for almost 30% of the test cases, whereas the POD provides the same accuracy for less than

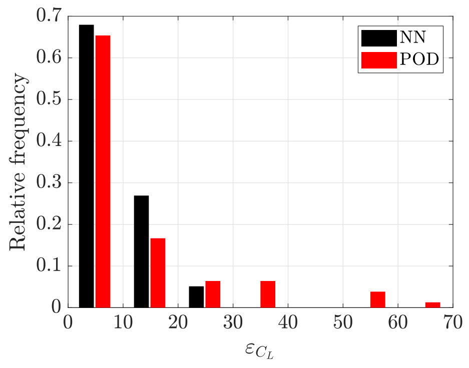


Figure 9: Relative frequency of the error on the test set, measured in lift counts, for the proposed NN and the POD.

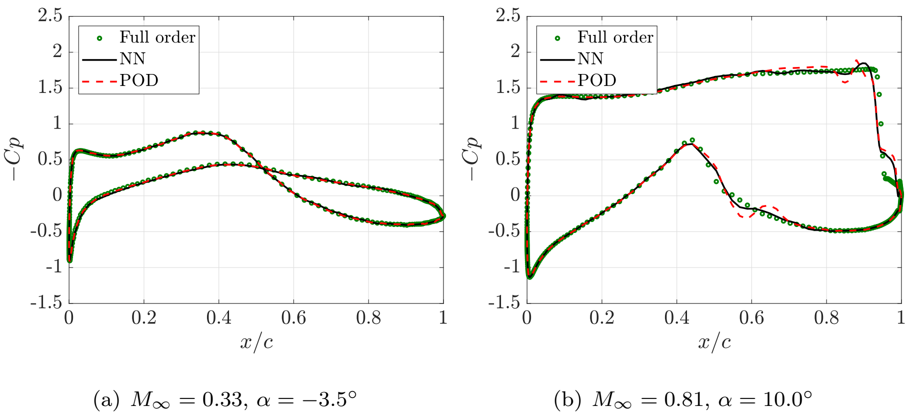


Figure 10: Comparison of the pressure coefficient, C_p, obtained with the CFD solver and the predicted C_p using NN and the POD.

20% of the test cases. In addition, the worst prediction for the NN is below 30 lift counts, whereas the POD provides an error on the lift higher than 60 lift counts.

To further illustrate the comparison on the performance of the proposed NN and the POD, figure 10 shows the plot of the pressure coefficient, C_p, obtained with the CFD solver and the predicted pressure distributions, C_p. The two different cases considered correspond to a subsonic and a transonic case. The results show that both methods provide an accurate prediction of the pressure

i	1	2	3	4	5	6	7	8
x_i	0.500	0.297	-0.003	-0.317	-0.437	-0.474	-0.496	-0.500
y_i^+	0.000	0.030	0.061	0.062	0.040	0.030	0.013	0.000

Table 1: Control points of the top curve of the base geometry approximating a NACA0012 aerofoil.

coefficient for the subsonic case. The prediction for the transonic case is substantially more challenging. The approaches employed show a deviation with respect to the reference CFD results in the region near the strong shock on the upper surface of the aerofoil. The POD shows a more pronounced oscillatory solution near the strong shock and also near the weaker shock on the lower surface around $x/c = 0.6$. This comparison shows the robustness of the NN near highly non-linear shock region.

5.3. Lift prediction on a geometrically parametrised aerofoil

The last example in two dimensions involves the prediction of the lift coefficient for an aerofoil that is parametrised using the control points of the NURBS curves describing the aerofoil. The base geometry corresponds to an approximation of the NACA0012 aerofoil using two cubic B-splines with eight control points each to define the top and bottom surface of the aerofoil respectively, as proposed in [61]. The knot vector of both NURBS is given by

$$\Lambda = \{0, 0, 0, 0, 0.615, 0.904, 0.941, 0.979, 1.019, 1.019, 1.019, 1.019\} \tag{22}$$

and the set of control points for the top curve $\{\mathbf{P}_i^+ = (x_i, y_i^+)\}_{i=1,\dots,8}$ are detailed in table 1. The control points for the bottom curve are symmetrically located with respect to the x axis and are defined as, $\mathbf{P}_i^- = (x_i, y_i^-)$ with $y_i^- = -y_i^+$, for $i = 1, \dots, 8$.

The end control points of both NURBS curves are fixed, i.e. $\mathbf{P}_1^- = \mathbf{P}_1^+ = (0.5, 0)$ and $\mathbf{P}_8^- = \mathbf{P}_8^+ = (-0.5, 0)$. In addition, the control point $\mathbf{P}_2^-$ is restricted to be aligned with $\mathbf{P}_1^- = \mathbf{P}_1^+$ and $\mathbf{P}_2^+$. More precisely, it is assumed that $\mathbf{P}_2^- = \mathbf{P}_1^+ + \mu(\mathbf{P}_1^+ - \mathbf{P}_2^+)$ with μ being a parameter defined in $[0.5, 1.5]$. This restriction ensures $\mathcal{G}^1$ continuity of the aerofoil geometry and leads to a problem with 25 independent geometric parameters. The variation of the position of the control points with respect to the base geometry, namely $(\pm\delta x_i, \pm\delta y_i)$ is considered to be the input of the neural network, where $(\delta x_i, \delta y_i) \in [0, 0.1c]^2$ and c denotes the chord of the aerofoil.

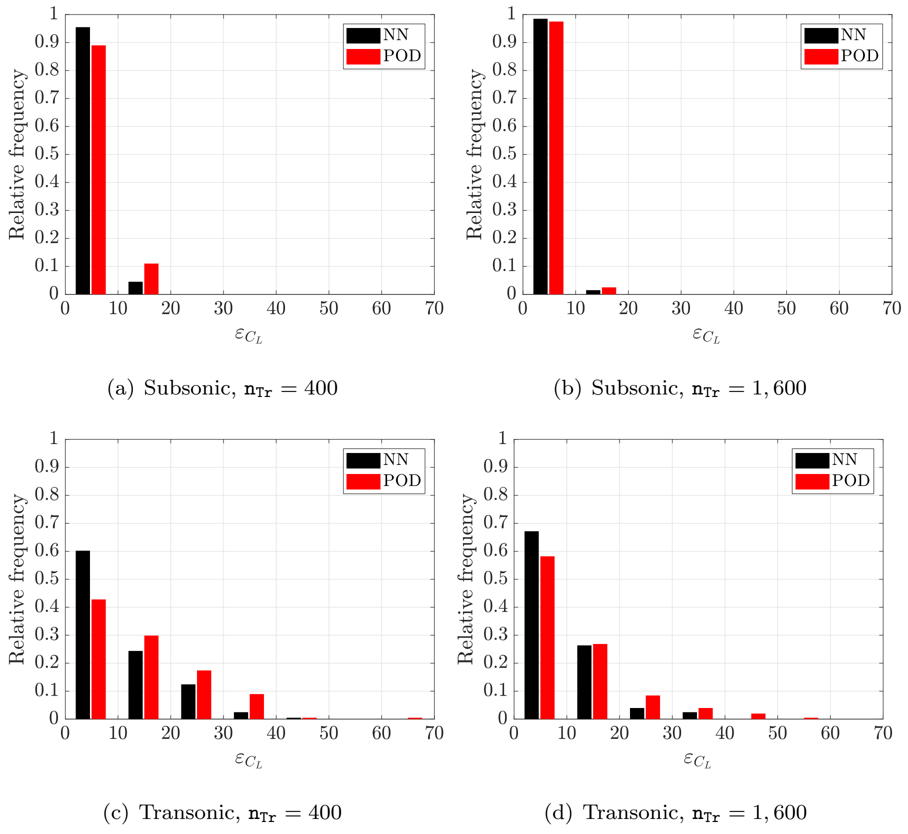
Two flow conditions are considered to explore the performance of the proposed NN and the POD approach for subsonic and transonic flows. The subsonic case corresponds to a free-stream Mach number $M_\infty = 0.4$ and an angle of attack $\alpha = 2^\circ$, whereas the transonic case corresponds to $M_\infty = 0.8$ and $\alpha = 2^\circ$.

The number of training cases, $\mathtt{n_{Tr}}$, varies from 400 to 1,600 and 200 test cases are considered. Both the training and test sets are obtained by using the latin hypercube sampling. Following the conclusions from the previous examples, the NN employs a single hidden layer with 98 neurons and the over-fitting parameter is selected as $\lambda = 0$. For the POD approach, all the snapshots are used in the interpolation. Contrary to the findings in the previous example, this option leads to the most accurate results here. This is attributed to the fact that the training cases do not contain a change of regime from subsonic to transonic flow.

Figure 11 compares the performance of the proposed NN and the POD with RBFs for both subsonic and transonic flows. The histograms compare the relative frequency of the error, measured in lift counts, for all the test cases. In all the experiments performed, the proposed NN outperforms the POD with RBFs.

For the subsonic example and using $\mathtt{n_{Tr}} = 400$ cases, the lift is predicted with an error below 10 lift counts in 96% of cases with the NN and in 89% of cases with the POD. When the number of training cases or snapshots is increased to $\mathtt{n_{Tr}} = 1,600$, the lift is predicted with an error below 10 lift counts in 99% of cases with the NN and in 98% of cases with the POD. In both cases, the worst predictions made by both the NN and the POD correspond to cases with an error below 20 lift counts.

The transonic example is much more challenging, as it requires an accurate prediction of the shock position. Using $\mathtt{n_{Tr}} = 400$ cases, the lift is predicted

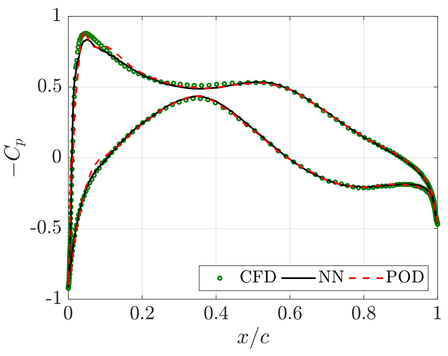


(a) Subsonic, $\mathtt{n_{Tr}} = 400$ (b) Subsonic, $\mathtt{n_{Tr}} = 1,600$

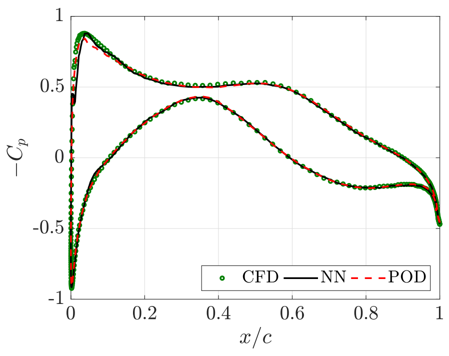
(c) Transonic, $\mathtt{n_{Tr}} = 400$ (d) Transonic, $\mathtt{n_{Tr}} = 1,600$

Figure 11: Relative frequency of the error on the test set, measured in lift counts, for the proposed NN and the POD for subsonic and transonic flows and with different number of training cases, $\mathtt{n_{Tr}}$.

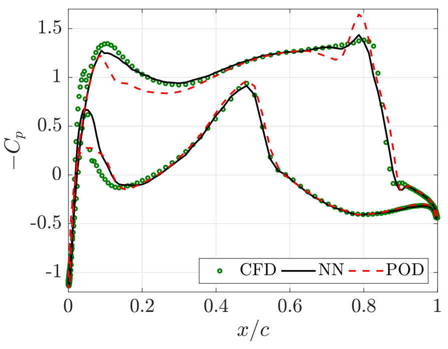
with an error below 10 lift counts in 60% of cases with the NN and in 43% of cases with the POD. The number of cases that are predicted with an error between 10 and 30 lift counts with both the NN and the POD are 37% and 47% respectively. The main difference between the NN and POD approaches is that, with the NN only 6 predictions have an error higher than 30 lift counts, whereas with the POD, 20 predictions have an error higher than 30 lift counts. In addition, with the POD the maximum error is 20 lift counts higher than the maximum error of the NN. Using $\mathtt{n_{Tr}} = 1,600$ cases, the lift is predicted with an error below 10 lift counts in 68% of cases with the NN and in 58% of cases



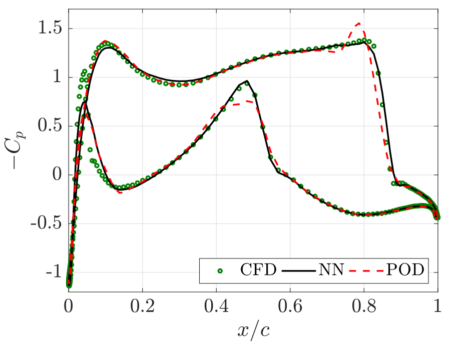
(a) Subsonic, $\mathtt{n_{Tr}} = 400$



(b) Subsonic, $\mathtt{n_{Tr}} = 1,600$



(c) Transonic, $\mathtt{n_{Tr}} = 400$



(d) Transonic, $\mathtt{n_{Tr}} = 1,600$

Figure 12: Pressure coefficient over the aerofoil of figure 13 compared to the predictions by the NN and the POD.

with the POD. Once more, the main difference is that the maximum error is 37 and 58 lift counts for the NN and the POD approaches respectively.

To illustrate the performance of the NN and POD approaches, figure 12 compares the predicted pressure distribution over the aerofoil configuration shown in figure 13 for a subsonic and a transonic case, both using the proposed NN and the POD and for different number of training cases. The results in figures 12(a) and 12(b) clearly show that both the NN and POD approaches are capable of producing accurate predictions of the pressure coefficient in a subsonic regime. For the transonic regime when $\mathtt{n_{Tr}} = 400$, there is considerable difference between the POD results and the reference results. This is particularly noticeable

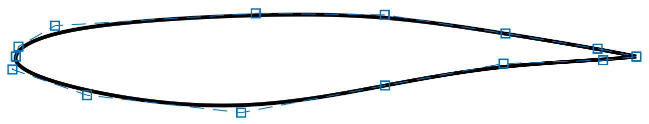


Figure 13: Geometry of the aerofoil chosen to compare the NN and POD prediction capability in figure 12. The squares denote the control points of the NURBS and the discontinuous line is the control polygon.

in the oscillatory character of the prediction near the strong shock on the upper curve. The discrepancy is also visible in the pressure coefficient distribution over the upper curve near the leading edge. The NN provides a better prediction of the shock without showing oscillations and it also produces a better approximation of the pressure coefficient distribution near the leading edge. Both the NN and the POD show some discrepancies with respect to the reference results for the pressure coefficient distribution over the lower curve near the leading edge. When the number of training cases is increased to $\texttt{n}_{\texttt{Tr}} = 1,600$, the POD still shows a poor approximation near the strong shock on the top curve as well as near the weaker shock in the bottom curve, whereas the NN shows a very good agreement with respect to the reference results.

6. Numerical examples in three dimensions

The multi-output NN outperformed the POD employed in all the examples considered in two dimensions. This shows the superiority of the proposed multi-output NN against the POD. As a result, only NNs of different configurations are employed in the examples in three dimensions.

6.1. Aerodynamic predictions on the ONERA M6 wing at various inflow conditions

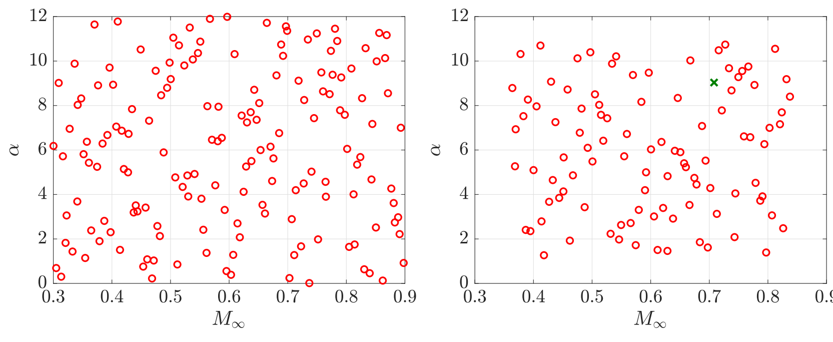
The first example considers the computation of the aerodynamic coefficients of the ONERA M6 wing [62] at a free stream Mach number, M_∞, and an angle of attack, α, in predefined intervals $I_M = (0.3, 0.9)$ and $I_\alpha = (0^\circ, 12^\circ)$, respectively. The range of the inflow conditions considered leads to both subsonic and transonic flows.

This example is used to compare the accuracy of the predictions made by three NNs. The first network considers M_∞ and α as inputs and one aerodynamic coefficient, the lift, drag or moment, as a single output. The second NN also considers M_∞ and α as inputs and the three aerodynamic coefficients as outputs. The third network considers the same inputs as the other two networks and the output is the pressure at a user defined set of points on the wing. The set of points considered corresponds to the mesh nodes used to discretise the wing.

The networks are trained using a dataset of $\mathtt{n_{Tr}} = 160$ simulations performed with the vertex-centred finite volume method [48]. Every trained network is then tested using a dataset of $\mathtt{n_{Te}} = 100$ simulations. Both dataset is selected by using the latin hypercube sampling [59] in $I_M \times I_\alpha$. Figure 14 shows the sampling space of the training and the test sets. In contrast to figures 2, more training data is used to ensure the sampling space is well covered. Moreover, the test set is defined using a smaller interval as $I_M = (0.35, 0.85)$ and $I_\alpha = (1^\circ, 11^\circ)$, so that there is no extrapolation. The volume mesh used in this example consists of 167,824 elements and 29,025 nodes. Figure 15 shows the surface mesh used in this analysis, which consists of 1,977 nodes and 3,900 triangles.

The first numerical experiment explores the accuracy of the predicted aerodynamic quantities as a function of the number of hidden neurons, $\mathtt{n_N}$, and the number of hidden layers, $\mathtt{n_L}$, for all the three networks considered and for different values of the over-fitting parameter, λ.

Figure 16 shows the evolution of the mean error in the lift as a function of the number of hidden layer, $\mathtt{n_L}$, and hidden neurons, $\mathtt{n_N}$. The results show that the accuracy of the first two networks is comparable, with an error between 10 to 15 lift counts in the majority of cases. The second network is able to provide higher accuracy, with an error near five lift counts, with one hidden layer and $\mathtt{n_N} = 180$. Finally, the third network provides the most accurate results, with an error below three lift counts for any combination of the values of $\mathtt{n_L}$ and $\mathtt{n_N}$.



(a) Training set (b) Test set

Figure 14: The sampling space used to define the training and test dataset using $\mathtt{n_{Tr}} = 160$ and $\mathtt{n_{Te}} = 100$ simulations respectively

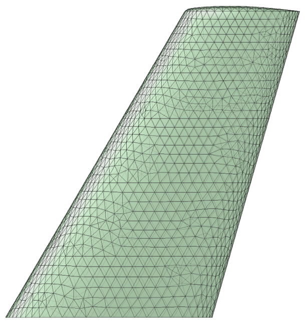
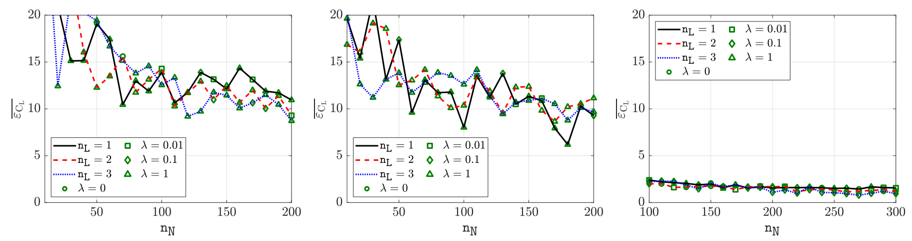


Figure 15: The surface mesh used to obtain CFD data in $I_M \times I_\alpha$, consisting of 1,977 nodes and 3,900 triangles.



(a) Network 1 (b) Network 2 (c) Network 3

Figure 16: Mean value of the lift counts, $\overline{\varepsilon_{C_L}}$, measured in the test set as a function of the number of hidden neurons, $\mathtt{n_N}$, and the number of hidden layers, $\mathtt{n_L}$.

This experiment shows the robustness of the third NN model.

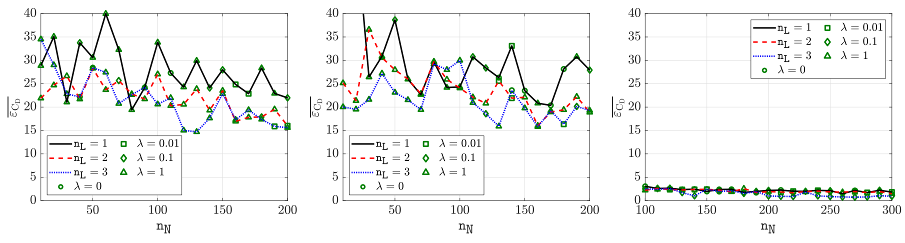

(a) Network 1 (b) Network 2 (c) Network 3

Figure 17: Mean value of the drag counts, $\overline{\varepsilon_{C_D}}$, measured in the test set as a function of the number of hidden neurons, $\mathtt{n_N}$, and the number of hidden layers, $\mathtt{n_L}$.

Figure 17 shows the mean value of the errors of the three types of trained NNs, measured in drag counts, as function of hidden layers, $\mathtt{n_L}$ and hidden neurons, $\mathtt{n_N}$. The first two networks are observed to have a higher level of oscillations, indicating a strong dependence on the number of hidden neurons and layers. The networks are not able to produce an accuracy lower than 15 drag counts, both achieved with three hidden layers and a comparatively larger number of neurons, namely $\mathtt{n_N} \in [120, 200]$. The third network provides again the most accurate results and it also shows a weak dependence on the hyperparameters.

Finally, figure 18 shows the mean value of the error, measured in moment counts, as a function of the number of hidden layers, $\mathtt{n_L}$ and the number of hidden neurons, $\mathtt{n_N}$ for the three different types of networks considered. Similar conclusions can be drawn as both the first and second networks are observed to have more oscillations with a maximum accuracy of no less than 50 counts for any combination of hyperparameters. In this figure, the superiority of the third network is clearer with a marginally lower error measured in the test set. The highest accuracy for this network leads to an error of only six moment counts, achieved with three hidden layers and $\mathtt{n_N} = 220$ neurons. This corresponds to the same configuration used to predict the lift and drag coefficients.

It is worth noting that the network proposed in this work provides greater accuracy for this three dimensional example, compared to the two dimensional

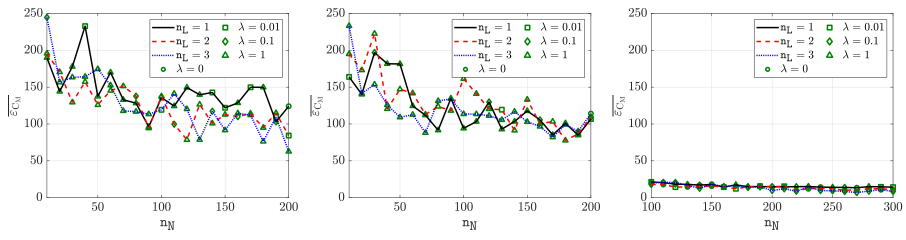

(a) Network 1 (b) Network 2 (c) Network 3

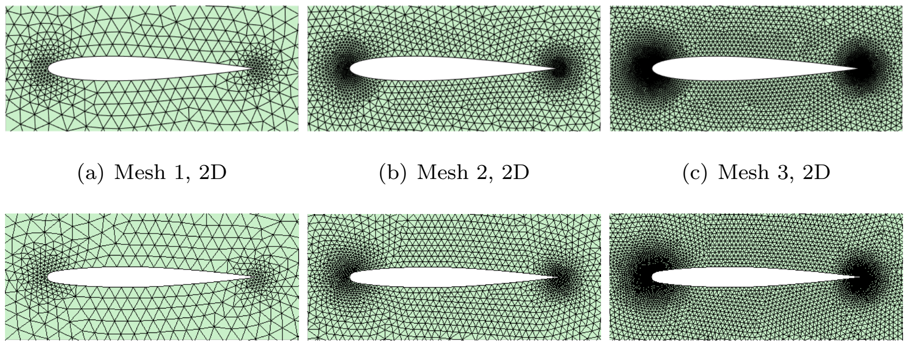
Figure 18: Mean value of the moment counts, $\overline{\varepsilon_{C_M}}$, measured in the test set as a function of the number of hidden neurons, $\mathtt{n_N}$, and the number of hidden layers, $\mathtt{n_L}$.

results of previous examples. To better understand this phenomenon, two studies are performed. First, the influence of the accuracy of the CFD data on the accuracy of the NN predictions is studied. Next, the flow features present in the two and three dimensional cases are analysed and compared to a three dimensional example that considers an extruded wing, rather than the Onera M6 swept wing. Only the third network is used because, as previously shown, it provides the most accurate results.

To study the influence of the accuracy of the CFD data on the accuracy of the NN predictions, three levels of mesh refinement are considered in two and three dimensions. Figure 19 shows the meshes employed in this example. There are approximately the same number of mesh nodes across any xz-plane in the spanwise direction of the ONERA M6 wing as in the corresponding meshes in two dimensions.

The two dimensional meshes have 1,936, 5,810 and 12,964 elements, and 51, 101 and 167 nodes are used to discretise the aerofoil surface, respectively. The three dimensional volume mesh used previously in this example, represents the coarse mesh. The medium and fine volume meshes in three dimensions have 1,304,444 and 4,485,190 elements, and 7,307 and 16,259 nodes, respectively.

For each mesh a set of $\mathtt{n_{Tr}} = 160$ training cases and a set of $\mathtt{n_{Te}} = 100$ test cases are generated. The pressure at the nodes used to discretise the aerofoil and wing, in two and three dimensions respectively, is used as the output of the


(a) Mesh 1, 2D (b) Mesh 2, 2D (c) Mesh 3, 2D

(d) Mesh 1, 3D (e) Mesh 2, 3D (f) Mesh 3, 3D

Figure 19: Detailed view of the three meshes used for the NACA 0012 aerofoil (top) and on the symmetry plane for the ONERA M6 wing (bottom).

NN model.

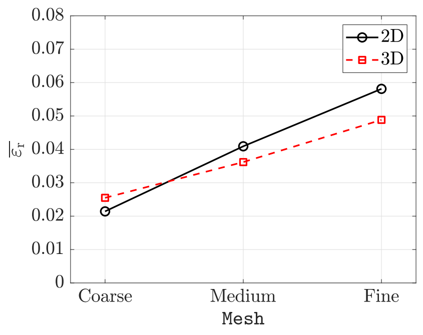
To measure the accuracy of the NNs, the relative error of the pressure coefficient, defined as

$$\varepsilon_{\mathtt{r}} = \left[\frac{\displaystyle\int_{\Gamma} \left(\left[C_p(\boldsymbol{x}^+) - C_p^\star(\boldsymbol{x}^+) \right]^2 + \left[C_p(\boldsymbol{x}^-) - C_p^\star(\boldsymbol{x}^-) \right]^2 \right) d\Gamma}{\displaystyle\int_{\Gamma} \left(\left[C_p(\boldsymbol{x}^+) \right]^2 + \left[C_p(\boldsymbol{x}^-) \right]^2 \right) d\Gamma} \right]^{1/2}, \tag{23}$$

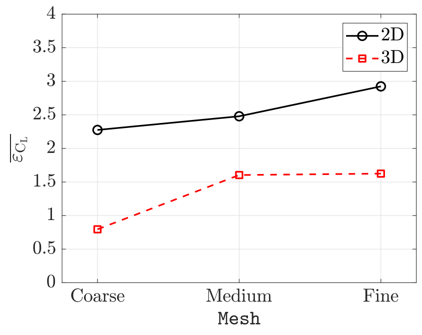
where $C_p(\boldsymbol{x}^+)$ and $C_p(\boldsymbol{x}^-)$ denote the target pressure coefficient distribution over the top and bottom surface describing the aerofoil/wing, $C_p^\star$ denotes the corresponding predicted pressure coefficient and Γ is the surface of the aerofoil/wing. In addition, the accuracy on the three aerodynamic quantities of interest is also considered. To offer a better comparison between the two and three dimensional results, the moment coefficient in three dimensions is computed about the aerodynamic centre of each section.

Figure 20 shows the comparison of the mean value of the errors in the test dataset. Every point in this figure represents the lowest error obtained by selecting the number of hidden layers, $\mathtt{n_L}$, the number of hidden neurons, $\mathtt{n_N}$ and the over-fitting parameter, λ, as done in previous experiments.

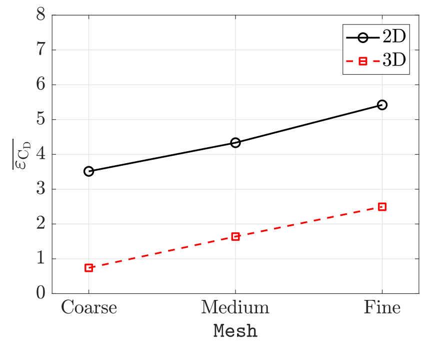
The results show that the accuracy of the networks when predicting values of the pressure over the aerodynamic shape is almost identical in two and three



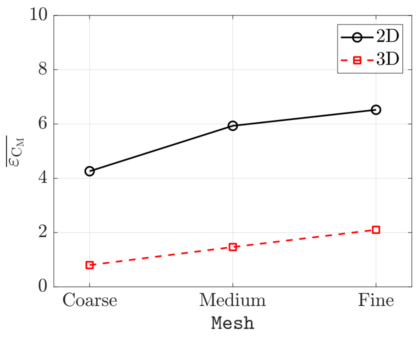
(a) Relative error in C_p



(b) Error in C_L



(c) Error in C_D

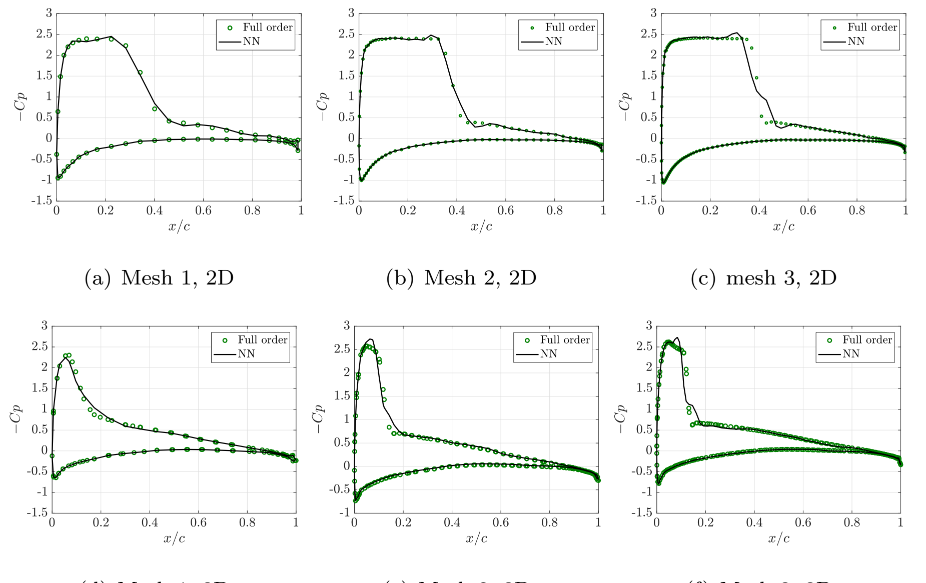


(d) Error in C_M

Figure 20: Mean value of the errors, measured in the test set, for the three levels of mesh refinement in two and three dimensions.

dimensions. Moreover, it is observed that the error in the prediction increases as the mesh is refined. This behaviour is attributed to the fact that coarser meshes tend to smooth sharp gradients in the solution, such as shocks. As shown in previous studies of sections 5.2 and 5.3, NN produces more accurate predictions when the flow field is smooth (e.g subsonic flow), compared to cases with sharp gradients, (e.g transonic flow).

To further illustrate the difference between the predictions in two and three dimensions, figure 21 shows a comparison of the pressure coefficient, C_p, obtained with the CFD solver and the predicted C_p using NN for a test case in the transonic regime, highlighted with a cross in figure 14(b). The results show



(a) Mesh 1, 2D (b) Mesh 2, 2D (c) mesh 3, 2D

(d) Mesh 1, 3D (e) Mesh 2, 3D (f) Mesh 3, 3D

Figure 21: Comparison of the pressure coefficient, C_p, obtained with the CFD solver and the predicted C_p using NN for a transonic test case at a free stream Mach number, $M_\infty = 0.71$ and an angle of attack, $\alpha = 9.1^\circ$, for the meshes employed in two and three dimensions.

how the shock is smeared leading to a smooth variation of the pressure that is accurately predicted by the NN. As the level of mesh refinement is increased, the shock becomes sharper and the prediction made by the NN presents some overshoots and undershoots near the shock. It can therefore be deduced that as the meshes become finer, the predicted solution from the trained models may contain oscillations near regions of sharp gradients, such as shocks. However this may or may not affect the results measured in the count metrics as it is a calculated integral computed on the surface of the aerofoil or wing.

When the error is measured in the lift, drag or moment, the results of figure 20 show that more accurate predictions are obtained in three dimensions. This behaviour is attributed to the different flow features induced by the two dimensional NACA0012 aerofoil (equivalent to an extruded three dimensional wing) and the Onera M6, which is a swept wing.

To confirm this hypothesis, the second level of mesh refinement is considered

Mesh	$\overline{\varepsilon_{\mathtt{r}}}$	$\overline{\varepsilon_{C_L}}$	$\overline{\varepsilon_{C_D}}$	$\overline{\varepsilon_{C_M}}$
2D	0.041	2.5	4.3	6.0
3D-X	0.042	2.6	4.4	6.1
3D	0.036	1.6	1.6	1.7

Table 2: Mean value of the errors, measured in the test dataset, for the medium level of mesh refinement when $\mathtt{n}_{\mathtt{Tr}} = 160$ simulations are used in the training of the third network.

and the NACA0012 profile is used to built an extruded wing in three dimensions. The generated mesh maintains the same spacing over the wing to ensure that the results between the extruded geometry and the two dimensional case can be compared.

Table 2 shows the comparison of the mean value of the errors obtained from three meshes when the same training and test datasets, as shown in figure 14, are employed. The results show that the accuracy of the predictions for the two dimensional case is almost identical to the three dimensional case with the extruded wing (3D-X). For the three dimensional swept wing, the accuracy is higher. A further analysis of the flow features indicates that the strength of the shock in the simulation with the Onera M6 wing is not as strong as in the two dimensional and three dimensional extruded wing. In addition, the higher accuracy obtained in the Onera M6 predictions is also attributed to that swept wings are known to delay the appearance of shocks. Hence, more cases in the datasets contain smoother solutions for the wide range of flow conditions considered in this example and thus lower mean values of the errors in the predictions are obtained.

6.2. Aerodynamic predictions on a geometrically parametrised wing

This example involves the prediction of the aerodynamic coefficients on a wing that is parametrised using the control points of the NURBS describing the surface. The base geometry corresponds to an approximation of the ONERA M6 wing [62] and consists of two surfaces, the top and bottom surface, each using six cubic B-splines with eight control points to define the surface. In this

i	1	2	3	4	5	6	7	8
$x_{i,1}$	0.0	0.0	31.8	136.9	332.6	571.0	738.1	805.9
$y_{i,1}$	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
$z^+_{i,1}$	0.0	9.5	26.3	34.6	43.9	26.6	8.4	0.0
$x_{i,6}$	690.6	690.6	708.5	767.6	877.7	1011.8	1105.8	1143.9
$y_{i,6}$	1196.3	1196.3	1196.3	1196.3	1196.3	1196.3	1196.3	1196.3
$z^+_{i,6}$	0.0	5.3	14.8	19.4	24.7	14.9	4.7	0.0

Table 3: Control points of the top curve at the root and tip aerofoil section of the base geometry approximating the ONERA M6 wing.

example, a planar surface is considered at the wingtip. The knot vectors of NURBS curves representing the surfaces are given by

$$\begin{aligned} \Lambda_u &= \{0, 0, 0, 0, 0.031, 0.126, 0.384, 0.749, 1, 1, 1, 1\} \\ \Lambda_v &= \{0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1\} \end{aligned} \tag{24}$$

and the set of control points for the top curve $\{\mathbf{P}^+_{i,j} = (x_{i,j}, y_{i,j}, z^+_{i,j})\}_{i=1,\dots,8,\ j=1,6}$ are detailed in table 3. The control points for the bottom surface are symmetrically located with respect to the xy-plane and are defined as, $\mathbf{P}^-_{i,j} = (x_{i,j}, y_{i,j}, z^-_{i,j})$ with $z^-_{i,j} = -z^+_{i,j}$, for $i = 1, \dots, 8$.

The control points on the leading and trailing edges are fixed, i.e. $\mathbf{P}^-_{1,j} = \mathbf{P}^+_{1,j}$ and $\mathbf{P}^-_{8,j} = \mathbf{P}^+_{8,j}$. In addition, the control point $\mathbf{P}^-_{2,j}$ is restricted to be aligned with $\mathbf{P}^-_{1,j} = \mathbf{P}^+_{1,j}$ and $\mathbf{P}^+_{2,j}$. More precisely, it is assumed that $\mathbf{P}^-_{2,j} = \mathbf{P}^+_{1,j} + \mu(\mathbf{P}^+_{1,j} - \mathbf{P}^+_{2,j})$ with μ being a parameter defined in $[0.5, 1.5]$. This restriction ensures $\mathcal{G}^1$ continuity at the leading edge of the wing and leads to a problem with 25 independent geometric parameters in one aerofoil section. Figure 22 shows the 96 control points in the parametrised geometry used to approximate the ONERA M6 wing. However, in this example, only the control points in the root and tip chord of the wing, coloured red in figure 22 are varied in the xz-plane, that is, $\delta y_{i,j} = 0$. This amounts to a total of 50 independent geometric parameters in the analysis. The variation of the positions of these control points

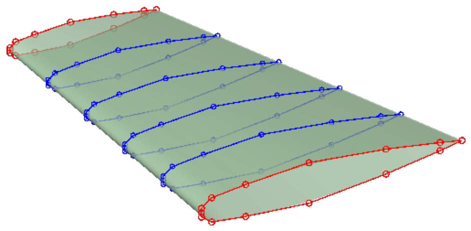


Figure 22: The ONERA M6 wing used as the base geometry and parametrised using the control points of the NURBS, denoted by circles. The lines represent the control polygon. The independent variables are coloured in red and the dependent variables in blue.

with respect to the base geometry, namely $(\pm\delta x_{i,j}, \pm\delta z_{i,j}))$, is considered to be the inputs to the neural network, where $(\delta x_{i,j}, \delta z_{i,j}) \in [0, 0.1c]^2$ and c denotes the chord of the aerofoil. To ensure that the variation of the control points does not lead to unrealistic geometries with very localised sudden change on the surface definition, the control points located in between the root and tip chord, coloured in blue in figure 22 are linearly displaced and follow the relationship

$$\boldsymbol{x}^*_{i,j} = \boldsymbol{x}_{i,j} + \delta\boldsymbol{x}_{i,1}(1 - \frac{y_{i,j}}{b}) + \delta\boldsymbol{x}_{i,6}\frac{y_{i,j}}{b}, \qquad i = 1, \dots, 8, j = 1, \dots, 6 \tag{25}$$

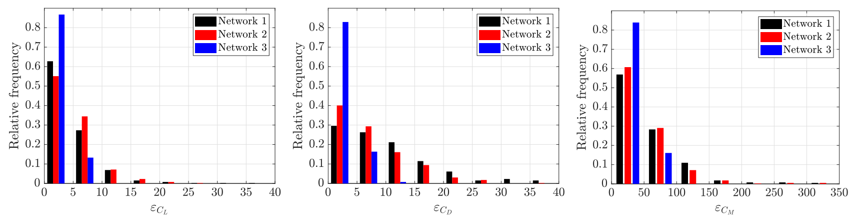
where $\boldsymbol{x}_{i,j}$ denotes the coordinate vector of i-th control points in the j-th aerofoil section, $\boldsymbol{x}^*_{i,j}$ represents the moved coordinate vector and b is the wingspan. The surface triangulation of the base geometry is generated using the advancing front method. The Cartesian coordinates of surface mesh nodes are projected to obtain the corresponding parametric coordinates of the NURBS surface. Given a new set of control points, obtained from design of experiment [59], the same parametric coordinates are evaluated to obtain a new set of Cartesian coordinates for the new geometry. This allows the designer to use the same number of nodes to discretise the geometry in the surface mesh. The coordinates on the planar wingtip surface and the symmetry boundary wall surface are then updated using the Laplacian smoothing technique [63]. A new volume mesh is generated for each new surface mesh using the Delaunay meshing scheme and the nodes are placed according to a specified background mesh and user defined sources [64]. There are 11,283 nodes used to discretise the wing geometry in the surface mesh and around 1.34 million elements in the volume meshes.

In this example, the training and test cases are of size $\mathtt{n_{Tr}} = 1600$ and $\mathtt{n_{Te}} = 400$ respectively. Both data sets are obtained by using the latin hypercube sampling. Steady Euler calculations are performed in the transonic regime at a free-stream Mach number, $M_\infty = 0.84$ and an angle of attack, $\alpha = 3.06°$.

Three types of neural networks, similar to the previous example are employed. All the three NNs consider the 50 geometric parameters representing the locations of the control points of the root and the tip aerofoil chord of the wing as inputs. The same three NN model described in section 6.1 are used.

To illustrate the resulting performance of the three NN employed, figure 23 quantifies the accuracies for the three aerodynamic quantities of interest, measured in counts. The histograms represent the relative frequency of the error for all test cases. In figure 23(a), the first NN employs a single hidden layer with $\mathtt{n_N} = 40$ and the over-fitting parameter is selected as $\lambda = 10$, the second NN employs a single hidden layer with 10 hidden neurons and the over-fitting parameter is selected as $\lambda = 1$, and finally the third NN employs one hidden layer with $\mathtt{n_N} = 290$ and the over-fitting parameter is set as $\lambda = 10$. The results show the first and second NNs predict only 63% and 56% of the test cases with an error below five lift counts respectively. The third NN, however, provides the most accurate predictions with an error below five lift counts for almost 85% of the test cases. This represents a significant difference of about 20%. Moreover, both the first and second NN are able to produce an error below 10 lift counts for almost 80% of the test cases, while the third NN is observed to achieve a marginally higher performance, with 100% of the test cases falling below 10 lift counts. The maximum error measured in the test dataset for the three NNs are 36, 26 and 9 lift counts respectively.

Figure 23(b) shows the relative frequency of the error, measured in drag counts. A different set of hyperparameters with the best accuracy is used for first NN, while the same configuration is maintained for the second and third NN. Here, the first NN employs a single hidden layer with 20 neurons and the over-fitting parameter is set as $\lambda = 10$. The third network produces a prediction



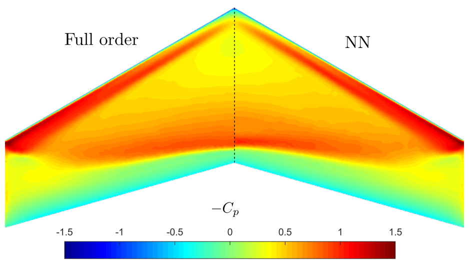
(a) Lift counts (b) Drag counts (c) Moment counts

Figure 23: Relative frequency of the error on the test set for the three neural networks employed, measured in counts.

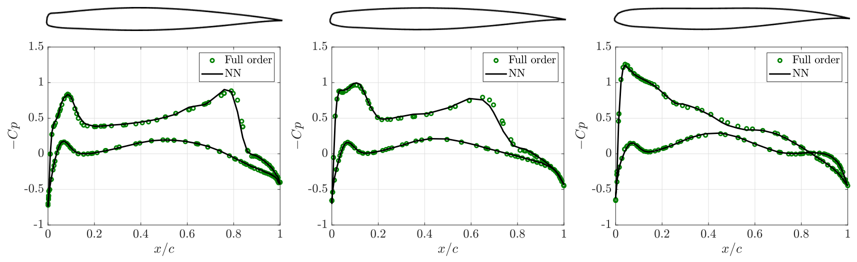
with an error below 5 drag counts for almost 83% of the test cases compared to the first and second network with only 30% and 40% of the cases, respectively. This represents a considerable difference of about 40% of the cases with better predictions. The maximum drag counts achieved for the three networks are 39, 38 and 11 drag counts, respectively.

Lastly, the histogram of figure 23(c) represents the relative frequency of the error, measured in moment counts. Here, the first two NNs have similar performance with almost 60% of the cases falling below 50 moments counts and about 25-30% of the cases below 100 moment counts. On the other hand, the third network provides a marginally higher accuracy of almost 85% of the cases below 50 moment counts and 100% below 100 moment counts. The maximum accuracies for the three networks employed are, 345, 326 and 96 moment counts. This shows the robustness of the third network for the use as an aerodynamic predictor for highly parametrised geometry.

Figure 24 shows a test case to further illustrate the comparison on the performance of the full order model and the output of the proposed NN model. Figure 24(a) shows the surface pressure contours of the full order and the corresponding NN prediction. The results show that, visually, there is no significant difference between the CFD data and the NN predictions. Figures 24(b), 24(c) and 24(d) show the aerofoil cross section and their corresponding pressure distributions for the full order and NN predictions at 20%, 50% and 80% in the spanwise direction, respectively. It can be observed how the geometry changes



(a) Surface pressure contour



(b) Predictions at 20% (c) Predictions at 50% (d) Predictions at 80%

Figure 24: Comparison of the pressure coefficients, C_p, obtained with the CFD solver and NN predictions using the third network for a test case at different locations in the spanwise direction of the wing.

across the cross section, from a comparatively blunt leading edge and an almost symmetric trailing edge at 20% to a more rounded leading edge, a flattened top and a comparatively higher cambered aft section at 80%, some of which are characteristics of supercritical aerofoils [65]. Figure 24(b) shows that NN predicts well and captures both shocks at the right position on the upper surface. Minor discrepancies in predicting the position of the second shock are observed in figure 24(c). Closer to the wingtip in figure 24(d), the reference result shows that there is no shock and the NN model is capable of capturing this behaviour well.

6.3. Aerodynamic predictions of a deforming wing

The last example considers the computation of the aerodynamic coefficients for a deforming wing at a free stream Mach number, M_∞, and an angle of attack,

α, in predefined intervals, $I_M = (0.5, 0.9)$ and $I_\alpha = (0^\circ, 10^\circ)$. The ONERA M6 wing is used as the baseline geometry and it undergoes a geometric twist along the y-axis and a wingtip deflection in the z-axis, in predefined intervals, $I_\beta = (0^\circ, 10^\circ)$ and $I_\kappa = (0 \text{ m}, 1 \text{ m})$. The range of inflow conditions considered leads to the subsonic and transonic transonic regime.

A geometric twist about the y-axis [66] can be written as

$$\boldsymbol{x}_i^\star = \boldsymbol{R}_y(\boldsymbol{x}_i - \boldsymbol{x}_i^{AC}) + \boldsymbol{x}_i^{AC}, \qquad \boldsymbol{R}_y = \begin{bmatrix} \cos\beta & 0 & -\sin\beta \\ 0 & 1 & 0 \\ \sin\beta & 0 & \cos\beta \end{bmatrix}, \tag{26}$$

for $i = 1, \dots, N$, where $\boldsymbol{x}_i$ represents a Cartesian coordinate on the wing, $\boldsymbol{x}_i^{AC}$ is the reference aerodynamic centre to which the rotation is performed and $\boldsymbol{x}_i^\star$ is the transformed coordinate. $\boldsymbol{R}_y$ denotes the rotation matrix about the y-axis, N is the total number of points used to discretise the geometry and β is the angle of twist, measured in degrees.

The wing is deflected from the xy-plane assuming the wing acts as a uniformly loaded cantilever beam, with one end fixed, that is the root of the wing, and a free end, here the wingtip. The deflection [67] can be written as

$$\delta z_i = \frac{\kappa}{3}\left[\frac{y_i^4}{b^4} - \frac{4y_i^3}{b^3} + \frac{6y_i^2}{b^2}\right], \tag{27}$$

for $i = 1, \dots, N$, where δz_i denotes the amplitude for i-th coordinate in the z direction, y_i is the initial length in the spanwise direction, b is the wingspan and κ is the maximum deflection at the wingtip. Moreover, it is ensured that for every transformed coordinate, the length in the spanwise direction is kept constant. The formulation of the arc length by integration is given as

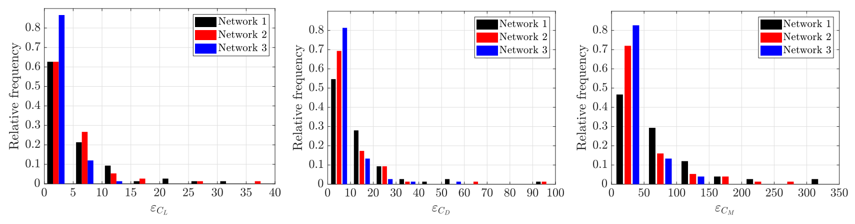
$$\tilde{b}_i = \int_0^{y_i^\star} \sqrt{1 + \left(\frac{\mathrm{d}z}{\mathrm{d}y_i}\right)^2}\,\mathrm{d}y, \tag{28}$$

for $i = 1, \dots, N$, where $\tilde{b}_i$ is the arc length of the i-th coordinate on the wing and is equal to y_i in the baseline geometry. $y_i^\star$ represents the new y coordinate for which the length is kept constant.

A reference volume mesh is generated around the ONERA M6 wing and the Delaunay graph method [68] is used to update the nodes of the transformed geometry. This mesh morphing scheme ensures that the primary mesh topology is maintained. Large displacements can be applied at the boundaries in a single step without the mesh entanglement that generally affect standard mesh movement procedures [68]. In this example, it was found that there is no negative volume and mesh entanglement moving from the primary mesh in one single step for a geometric twist of up to $\beta = 11°$ and a maximum deflection of $\kappa = 1.1$m at the wingtip.

The latin hypercube sampling is used to design the training and test dataset of size $\mathtt{n}_{\mathtt{Tr}} = 100$ and $\mathtt{n}_{\mathtt{Te}} = 75$, respectively. The reference volume mesh consists of 1,071,457 elements and 16,240 nodes are used to discretise the geometry. In this example, the three different types of neural network employed considers M_∞, α, β, and κ, as inputs. The first network has one single aerodynamic output, the lift, drag or moment coefficient. The second network has three aerodynamic coefficients as outputs and finally the third network considers the pressure defined at the 16,240 nodes used to discretise the geometry in the volume mesh, as outputs. As in previous examples, numerical experiment is performed to explore the accuracy of the predicted aerodynamics coefficients, measured in counts in the test set, as a function of the number of hidden layers, $\mathtt{n}_{\mathtt{L}}$, the number of hidden neurons, $\mathtt{n}_{\mathtt{N}}$, and various values of the over-fitting parameter, λ for the three types of NN considered.

Figure 25 shows the comparison of the relative frequency of the error, measured in counts in the test dataset for the three types of NNs employed. Figure 25(a) shows the relative frequency of the error measured in the prediction of the lift. Following the numerical experiment carried out in this example, the first network is selected to have three hidden layers and $\mathtt{n}_{\mathtt{N}} = 140$, the second NN has three hidden layers and 190 hidden neurons, and finally the third network employs one hidden layer with $\mathtt{n}_{\mathtt{N}} = 270$. All the selected network configurations employ an over-fitting parameter of $\lambda = 0.01$. The same configuration is maintained in the analysis of the drag and moment for the second and third

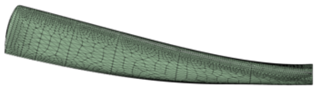


(a) Lift counts (b) Drag counts (c) Moment counts

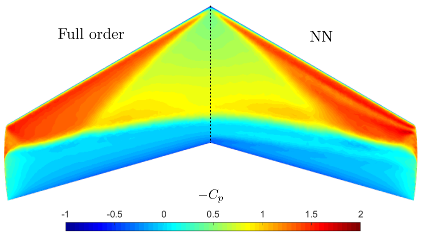
Figure 25: Relative frequency of the error on the test set for the three neural networks employed, measured in counts.

networks. The results show that both the first and second NNs achieve about 60% of the test cases below five lift counts while the third network achieves a marginal 88% of the test cases in the same range. The first noticeable difference is that the third network has almost 100% of the cases with an error less than 10 lift counts in the test set and the maximum error obtained with this NN is 11 lift counts. The second and third network provide a prediction with a maximum error of 33 and 36 lift counts, respectively. Figure 25(b) represents the relative frequency of the error, measured in drag counts. The first network provides a much lower accuracy of 55% of the cases below 10 drag counts, as opposed to 70% and 82% for the second and third networks. The three NNs have a maximum error of 98, 94 and 51 drag counts, all of which corresponds to the same case in the test dataset. Lastly, Figure 25(c) describes the relative frequency of the error in the test set, measured in the test set. The first NN achieves 46% of the cases below 50 moment counts, which is comparatively lower to the second and third NN, with 72% and 83% of the cases. The third network has a maximum error of 127 moment counts, while the first and second NNs provide an error of more than double of that, 400 and 270 moment counts respectively.

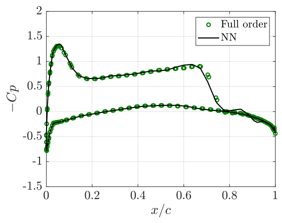
Figure 26 illustrates the comparison between the performance of the full order model and the output of the chosen NN model. Figure 26(a) shows the deformed geometry of the test case, with an amplification factor of 10. The CFD simulation corresponds to a free-stream Mach number, $M_\infty = 0.83$ and an angle of attack, $\alpha = 5.4°$, which leads to a transonic flow. Figure 26(b) shows



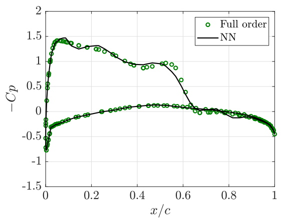
(a) $\beta = 3.0°$, $\kappa = 0.86$m



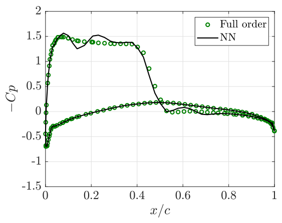
(b) Surface pressure contour



(c) Predictions at 20%



(d) Predictions at 50%



(e) Predictions at 80%

Figure 26: Comparison of the pressure coefficients, C_p, obtained with the CFD solver and NN predictions using the third network for a test case with inflow conditions, $M_\infty = 0.83$ and $\alpha = 5.4°$, at different locations in the spanwise direction of the wing.

the surface pressure contours of the full order on the left and the corresponding NN predictions on the right. The surface contour plots compare well with each other visually. Figures 26(c), 26(d) and 26(e) show the comparison of the surface pressure distributions for the full order and NN at 20%, 50% and 80% of the wing in the spanwise direction, respectively. Figure 26(c) shows that NN predicts well and captures both shocks at the right position on the upper surface. Minor discrepancies are observed over the top surface, at $x/c = 0.9c$. More oscillations are observed in the prediction over the top surface in both the 50% and 80% sectional pressure plots. The NN model captures the shocks relatively well and shows good agreement with the reference CFD results.

7. Concluding remarks

The paper presents a new multi-output NN methodology for the prediction of aerodynamic coefficients of aerofoils in two dimensions and wings in three dimensions. The proposed multi-output NN predicts the pressure distribution on the surface of aerofoils and wings by using inviscid compressible data from a vertex-centred CFD solver. The aerodynamic coefficients are computed after a pressure distribution is predicted.

The accuracy of the predictions using the proposed multi-output NN has been shown to be higher when compared to a NN where the three aerodynamic coefficients are predicted. The results also show the superiority when compared to a NN where only one aerodynamic coefficient is predicted. The superior performance has been demonstrated for both two and three dimensional examples involving free stream Mach number and the angle of attack as the inputs of the NNs.

In addition, the proposed NN has also been compared to the POD and the results shown the better performance of the NN, specially when predicting flows that involve shocks. The superior accuracy of the proposed NN has been shown for problems where the parameters involve flow conditions and for more challenging scenarios where the parameters are the control points of the NURBS that describe the geometry of the aerodynamic shape.

An extensive set of numerical studies has been presented to show the effect of the NN hyperparameters and numerical parameters of the POD on the accuracy of the predictions. Furthermore, the effect of increasing the number of cases in the training set in the accuracy of both the NNs and the POD has also been studied. The

The potential of the proposed NN has also been demonstrated for three dimensional examples involving flow conditions and geometric parameters. It is worth noting that examples involving up to 50 parameters have been considered in three dimensions. The influence of the accuracy of the CFD data into the accuracy of the NN predictions has been studied in two and three dimensions. It was found that the higher the accuracy in the CFD data, the more challenging is for the NNs to perform accurate predictions. This is attributed to the stronger shocks present in finer meshes due to the increased resolution of the flow field.

The accuracy of the predictions in two and three dimensions have also been compared. It has been shown that for a swept wing the NN offer a greater accuracy when compared to a two dimensional aerofoil. This is explained by the lower strength of the shocks in three dimensions and the delayed appearance of shocks in swept wings, leading to a greater set of training cases containing smooth solutions.

The proposed NN enables the fast prediction of aerodynamic quantities of interest and can be applied to speed up the design and optimisation of aerodynamic shapes.

Acknowledgements

The authors are grateful for the financial support provided by the Engineering and Physical Sciences Research Council (EP/P033997/1).

References

[1] M. La Mantia, P. Dabnichki, Effect of the wing shape on the thrust of flapping wing, Applied Mathematical Modelling 35 (10) (2011) 4979–4990.

[2] A. Ribeiro, A. Awruch, H. Gomes, An airfoil optimization technique for wind turbines, Applied Mathematical Modelling 36 (10) (2012) 4898–4907.

[3] J. Kou, W. Zhang, Multi-fidelity modeling framework for nonlinear unsteady aerodynamics of airfoils, Applied Mathematical Modelling 76 (2019) 832–855.

[4] L. Shi, Z. Han, M. Shahbaz, W.-P. Song, Surrogate-based robust airfoil design under aleatory operating-conditions and geometric uncertainties, in: Proceedings of the 54th AIAA aerospace sciences meeting, 2016, pp. 2016–0810.

[5] X. Wu, W. Zhang, S. Song, Uncertainty quantification and sensitivity analysis of transonic aerodynamics with geometric uncertainty, International Journal of Aerospace Engineering 2017 (2017).

[6] A. Quarteroni, G. Rozza, et al., Reduced order methods for modeling and computational reduction, Vol. 9, Springer, 2014.

[7] G. Rozza, D. B. P. Huynh, A. T. Patera, Reduced basis approximation and a posteriori error estimation for affinely parametrized elliptic coercive partial differential equations, Archives of Computational Methods in Engineering 15 (3) (2008) 229–275.

[8] G. Berkooz, P. Holmes, J. L. Lumley, The proper orthogonal decomposition in the analysis of turbulent flows, Annual review of fluid mechanics 25 (1) (1993) 539–575.

[9] K. Willcox, J. Peraire, Balanced model reduction via the proper orthogonal decomposition, AIAA journal 40 (11) (2002) 2323–2330.

[10] T. Lieu, C. Farhat, M. Lesoinne, POD-based aeroelastic analysis of a complete F-16 configuration: ROM adaptation and demonstration, in: 46th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference, 2005, p. 2295.

[11] S. Walton, O. Hassan, K. Morgan, Reduced order modelling for unsteady fluid flow using proper orthogonal decomposition and radial basis functions, Applied Mathematical Modelling 37 (20-21) (2013) 8930–8945.

[12] F. Ballarin, A. D'Amario, S. Perotto, G. Rozza, A POD-selective inverse distance weighting method for fast parametrized shape morphing, International Journal for Numerical Methods in Engineering 117 (8) (2019) 860–884.

[13] B. Schölkopf, A. Smola, K.-R. Müller, Kernel principal component analysis, in: International conference on artificial neural networks, Springer, 1997, pp. 583–588.

[14] R. Rosipal, M. Girolami, L. J. Trejo, A. Cichocki, Kernel pca for feature extraction and de-noising in nonlinear regression, Neural Computing & Applications 10 (3) (2001) 231–243.

[15] F. Chinesta, E. Cueto, A. Huerta, PGD for solving multidimensional and parametric models, in: Separated representations and PGD-based model reduction, Vol. 554 of CISM Courses and Lectures, Springer, Vienna, 2014, pp. 27–89.

[16] F. Chinesta, R. Keunings, A. Leygue, The proper generalized decomposition for advanced numerical simulations. A primer, Springer Briefs in Applied Sciences and Technology, Springer, Cham, 2014.

[17] R. Sevilla, S. Zlotnik, A. Huerta, Solution of geometrically parametrised problems within a CAD environment via model order reduction, Computer Methods in Applied Mechanics and Engineering 358 (2020) 112631.

[18] X. Zou, M. Conti, P. Díez, F. Auricchio, A nonintrusive proper generalized decomposition scheme with application in biomechanics, International Journal for Numerical Methods in Engineering 113 (2) (2018) 230–251.

[19] A. Leon, S. Mueller, P. de Luca, R. Said, J.-L. Duval, F. Chinesta, Non-intrusive proper generalized decomposition involving space and parameters: application to the mechanical modeling of 3D woven fabrics, Advanced Modeling and Simulation in Engineering Sciences 6 (1) (2019) 13.

[20] V. Tsiolakis, M. Giacomini, R. Sevilla, C. Othmer, A. Huerta, Nonintrusive proper generalised decomposition for parametrised incompressible flow problems in OpenFOAM, Computer Physics Communications In press. (2020) 107013.

[21] K. Elsayed, C. Lacor, CFD modeling and multi-objective optimization of cyclone geometry using desirability function, artificial neural networks and genetic algorithms, Applied Mathematical Modelling 37 (8) (2013) 5680–5704.

[22] M. Frank, D. Drikakis, V. Charissis, Machine-learning methods for computational science and engineering, Computation 8 (1) (2020) 15.

[23] D. J. Linse, R. F. Stengel, Identification of aerodynamic coefficients using computational neural networks, Journal of Guidance, Control, and Dynamics 16 (6) (1993) 1018–1025.

[24] S. Huang, L. Miller, J. Steck, An exploratory application of neural networks to airfoil design, in: 32nd Aerospace Sciences Meeting and Exhibit, 1994, p. 501.

[25] M. Raissi, P. Perdikaris, G. E. Karniadakis, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, Journal of Computational Physics 378 (2019) 686–707.

[26] M. Raissi, A. Yazdani, G. E. Karniadakis, Hidden fluid mechanics: A navier-stokes informed deep learning framework for assimilating flow visualization data, arXiv preprint arXiv:1808.04327 (2018).

[27] N. Asproulis, D. Drikakis, An artificial neural network-based multiscale method for hybrid atomistic-continuum simulations, Microfluidics and nanofluidics 15 (4) (2013) 559–574.

[28] E. Ulu, R. Zhang, L. B. Kara, A data-driven investigation and estimation of optimal topologies under variable loading configurations, Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization 4 (2) (2016) 61–72.

[29] J. S. Hesthaven, S. Ubbiali, Non-intrusive reduced order modeling of nonlinear problems using neural networks, Journal of Computational Physics 363 (2018) 55–78.

[30] R. Swischuk, L. Mainini, B. Peherstorfer, K. Willcox, Projection-based model reduction: Formulations for physics-based machine learning, Computers & Fluids 179 (2019) 704–717.

[31] L. Cao, K. S. Chua, W. Chong, H. Lee, Q. Gu, A comparison of pca, kpca and ica for dimensionality reduction in support vector machine, Neurocomputing 55 (1-2) (2003) 321–336.

[32] B. Schölkopf, S. Mika, A. Smola, G. Rätsch, K.-R. Müller, Kernel pca pattern reconstruction via approximate pre-images, in: International Conference on Artificial Neural Networks, Springer, 1998, pp. 147–152.

[33] B. Scholkopf, S. Mika, C. J. Burges, P. Knirsch, K.-R. Muller, G. Ratsch, A. J. Smola, Input space versus feature space in kernel-based methods, IEEE transactions on neural networks 10 (5) (1999) 1000–1017.

[34] M. Rai, Three-dimensional aerodynamic design using artificial neural networks, in: 40th AIAA Aerospace Sciences Meeting & Exhibit, 2002, p. 987.

[35] H. Sobieczky, Geometry generator for cfd and applied aerodynamics, in: New Design Concepts for High Speed Air Transport, Springer, 1997, pp. 137–157.

[36] M. Santos, B. Mattos, R. Girardi, Aerodynamic coefficient prediction of airfoils using neural networks, in: 46th AIAA aerospace sciences meeting and exhibit, 2008, p. 887.

[37] M. Khurana, H. Winarto, A. Sinha, Application of swarm approach and artificial neural networks for airfoil shape optimization, in: 12th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, 2008, p. 5954.

[38] G. Sun, Y. Sun, S. Wang, Artificial neural network based inverse design: Airfoils and wings, Aerospace Science and Technology 42 (2015) 415–428.

[39] A. Kharal, A. Saleem, Neural networks based airfoil generation for a given cp using bezier–parsec parameterization, Aerospace science and Technology 23 (1) (2012) 330–344.

[40] Y. Zhang, W. J. Sung, D. N. Mavris, Application of convolutional neural network to predict airfoil lift coefficient, in: 2018 AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, 2018, p. 1903.

[41] V. Sekar, M. Zhang, C. Shu, B. C. Khoo, Inverse design of airfoil using a deep convolutional neural network, AIAA Journal 57 (3) (2019) 993–1003.

[42] M. M. Rai, N. K. Madavan, Aerodynamic design using neural networks, AIAA journal 38 (1) (2000) 173–182.

[43] S. Suresh, S. Omkar, V. Mani, T. G. Prakash, Lift coefficient prediction at high angle of attack using recurrent neural network, Aerospace Science and Technology 7 (8) (2003) 595–602.

[44] N. R. Secco, B. S. Mattos, Artificial neural networks applied to airplane design, in: 53rd AIAA Aerospace Sciences Meeting, 2015, p. 1013.

[45] L. Piegl, W. Tiller, The NURBS Book, Springer Berlin Heidelberg, 1995.

[46] K. Morgan, J. Peraire, J. Peiro, O. Hassan, The computation of three-dimensional flows using unstructured grids, Computer Methods in Applied Mechanics and Engineering 87 (2-3) (1991) 335–352.

[47] N. Kroll, J. K. Fassbender (Eds.), MEGAFLOW - Numerical Flow Simulation for Aircraft Design, Springer Berlin Heidelberg, 2005.

[48] K. A. Sørensen, O. Hassan, K. Morgan, N. P. Weatherill, A multigrid accelerated hybrid unstructured mesh method for 3D compressible turbulent flow, Computational Mechanics 31 (1-2) (2003) 101–114.

[49] M. Hagan, H. Demuth, M. Beale, O. DeJesus, Neural network design, 2nd edition, Martin Hagan, 2014.

[50] A. Krogh, J. A. Hertz, A simple weight decay can improve generalization, in: Advances in neural information processing systems, 1992, pp. 950–957.

[51] H. Zhao, Y.-H. H. Tsai, R. R. Salakhutdinov, G. J. Gordon, Learning neural networks with adaptive regularization, in: Advances in Neural Information Processing Systems, 2019, pp. 11389–11400.

[52] D. E. Rumelhart, G. E. Hinton, R. J. Williams, Learning representations by back-propagating errors, Nature 323 (6088) (1986) 533–536.

[53] R. P. Lippmann, Anintroduction to computing with neural nets, IEEE Assp magazine 4 (2) (1987) 4–22.

[54] M. Siddique, M. Tokhi, Training neural networks: backpropagation vs. genetic algorithms, in: IJCNN'01. International Joint Conference on Neural Networks. Proceedings (Cat. No. 01CH37222), Vol. 4, IEEE, 2001, pp. 2673–2678.

[55] I. Sutskever, J. Martens, G. Dahl, E. Geoffrey, On the importance of initialisation and momentum in deep learning, in: In Proceedings of the 30th international conference on machine learning (ICML-13), Vol. 28, Atlanta, 2013, pp. 1139–1147.

[56] X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, in: Y. W. Teh, M. Titterington (Eds.), Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, Vol. 9 of Proceedings of Machine Learning Research, 2010, pp. 249–256.

[57] P. LeGresley, J. Alonso, Airfoil design optimization using reduced order models based on proper orthogonal decomposition, in: Fluids 2000 conference and exhibit, 2000, p. 2545.

[58] T. Bui-Thanh, M. Damodaran, K. E. Willcox, Aerodynamic data reconstruction and inverse design using proper orthogonal decomposition, AIAA journal 42 (8) (2004) 1505–1516.

[59] M. D. Mckay, R. J. Beckman, W. J. Conover, A comparison of three methods for selecting values of input variables in the analysis of output from a computer code, Technometrics 42 (1) (2000) 55–61.

[60] A. Chatterjee, An introduction to the proper orthogonal decomposition, Current science (2000) 808–817.

[61] R. Sevilla, S. Fernández-Méndez, Numerical integration over 2D NURBS-shaped domains with applications to NURBS-enhanced FEM, Finite Elements in Analysis and Design 47 (10) (2011) 1209–1220.

[62] J. W. Slater, Nparc alliance verification and validation archive, onera m6 wing (2015).
URL `https://www.grc.nasa.gov/WWW/wind/valid/m6wing/m6wing.html`

[63] L. R. Herrmann, Laplacian-isoparametric grid generation scheme, Journal of the Engineering Mechanics Division 102 (5) (1976) 749–907.

[64] N. P. Weatherill, O. Hassan, Efficient three-dimensional Delaunay triangulation with automatic point creation and imposed boundary constraints, International Journal for Numerical Methods in Engineering 37 (12) (1994) 2005–2039.

[65] C. D. Harris, Nasa supercritical airfoils: A matrix of family-related airfoils (1990).

[66] G. G. Slabaugh, Computing euler angles from a rotation matrix, Retrieved on August 6 (2000) (1999) 39–63.

[67] R. C. Hibbeler, S. Fan, Statics and mechanics of materials, Vol. 2, Prentice Hall Upper Saddle River, 2004.

[68] X. Liu, N. Qin, H. Xia, Fast dynamic grid deformation based on delaunay graph mapping, Journal of Computational Physics 211 (2) (2006) 405–423.