Anisotropic mesh spacing prediction using neural networks

Callum Lock, Oubay Hassan, Ruben Sevilla*, Jason Jones

[a]*Zienkiewicz Institute for Modelling, Data and AI, Faculty of Science and Engineering, Swansea Univeristy, Swansea, SA1 8EN, Wales, UK*

Abstract

This work presents a framework to predict near-optimal anisotropic spacing functions suitable to perform simulations with unseen operating conditions or geometric configurations. The strategy consists of utilising the vast amount of high-fidelity data available in industry to compute a target anisotropic spacing and train an artificial neural network to predict the spacing for unseen scenarios. The trained neural network outputs the metric tensor at the nodes of a coarse background mesh that is then used to generate meshes for unseen cases. Examples are used to demonstrate the effect of the network hyperparameters and the training dataset on the accuracy of the predictions. The potential of the method is demonstrated for examples involving up to 11 geometric parameters on CFD simulations involving a full aircraft configuration.

Keywords: Mesh generation, Spacing function, Machine learning, Near-optimal mesh prediction, Computational fluid dynamics

*Corresponding author

Email addresses: C.D.Lock@Swansea.ac.uk (Callum Lock), O.Hassan@Swansea.ac.uk (Oubay Hassan), R.Sevilla@Swansea.ac.uk (Ruben Sevilla), J.W.Jones@Swansea.ac.uk (Jason Jones)

1. Introduction

Unstructured mesh generation plays a critical role in numerical simulations, influencing both computational efficiency and solution accuracy. Traditional unstructured mesh generation techniques often rely on heuristic-based refinement criteria and expert-driven adjustments, which can be computationally expensive and time-consuming Dawes et al. (2001); Slotnick et al. (2014); Karman et al. (2017). In fact, the generation of unstructured meshes for complex geometries continues to be one of the most time-consuming stages in the simulation process. The problem is exacerbated when multiple simulations for different operating conditions or geometric configurations are required, as usual in a design or optimisation cycle.

One obvious alternative when the same geometry is to be analysed for different operating conditions is to generate a single mesh, fine enough to capture all the features for all the solutions to be computed Michal (2019). This practice was analysed in Lock et al. (2023) in terms of computational time and carbon emissions associated to the use of HPC facilities and it was concluded that it can lead to 35 times more carbon emissions when compared to running simulations on meshes tailored for each simulation.

Another alternative is to use automatic mesh adaptivity George et al. (2017). These methods require only an initial mesh, which is iteratively refined based on error estimation. However, the effectiveness of the adaptation depends on the initial mesh capturing key solution features. Moreover, single-entity error indicators often miss complex phenomena, necessitating multiple indicators that incorporate both primitive and derived variables to detect features like shock waves, contact discontinuities, flow separation, and turbulence-related structures. In addition, for complex three-dimensional simulations, the full adaptation process can involve 20 to 30 iter-

ations Loseille et al. (2010), with each cycle requiring solution computation, error estimation, mesh refinement, and solution interpolation.

Recent advancements in machine learning, particularly neural networks, have demonstrated their potential to enhance mesh generation by automating key aspects of the process. The use of machine learning to assist the mesh generation process was originally proposed in Chang-Hoi et al. (1991) but it was not until the early 2000s that the first examples involving three dimensional examples emerged Alfonzetti (2003), still involving simple geometries and in the context of electromagnetic problems. More recently the use of machine learning techniques for mesh generation and adaptation has received increasing attention.

In Zhang et al. (2021), artificial neural networks (ANNs) were used to predict spacing at a given location based on parameters related to partial differential equations, geometry, and boundary conditions. A similar approach for computational fluid dynamics (CFD) was proposed in Huang et al. (2021), where spacing from adaptively refined meshes is mapped onto a Cartesian grid, converted into a greyscale image, and used to train an ANN. These methods relate to the present work as they predict spacing functions for mesh generation in unseen cases. Other studies have explored ANNs for mesh and degree adaptivity Yang et al. (2022); Wallwork et al. (2022); Tlales et al. (2022), predicting near-optimal spacing for complex CFD applications Lock et al. (2023, 2024), and estimating mesh anisotropy Fidkowski and Chen (2021).

In this work a new approach is presented to predict an anisotropic spacing function for new simulations involving either new operating conditions or new geometric configurations. The strategy follows the rationale introduced in Lock et al. (2024) as it uses a background mesh to define a discrete spacing field. However, the current work extends the previous work to enable the prediction of a metric tensor rather

than predicting a scalar.

First, a strategy is proposed to compute the metric tensor at each point of a computational mesh that was used to perform a high-fidelity simulation. Second, a method to transfer the anisotropic spacing to a coarse background mesh is presented. The strategy introduces mesh morphing based on a classical Delaunay graph to ensure that the spacing can be transferred to the same background mesh, even when the simulations are performed for different geometries. Finally, a new ANN is proposed to enable the prediction of the metric tensor at each point of the background mesh for new, unseen, flow conditions or geometric configurations. The proposed ANN architecture exploits the mathematical properties of the metric tensors to ensure that the prediction leads to a valid metric tensor and also to minimise the required amount of outputs to be predicted.

Numerical results involve three dimensional examples in the context of inviscid compressible flows with variable flow conditions and geometric parameters. The examples, involving up to 11 geometric parameters demonstrate the potential and the accuracy of the proposed approach to define a near-optimal initial mesh that is suitable to perform simulations. It is worth noting that the present approach can be seen as a way to obtain a good initial mesh for new simulations and, if necessary, it can be integrated in a mesh adaptive cycle. If used in this context, the predicted mesh is expected to require very few, if any, adaptivity cycles to reach the required accuracy. In fact, numerical examples included in this work assess the suitability of the meshes to perform simulations.

The remainder of the paper is organised as follows. Section 2 presents the necessary concepts to introduce the proposed approach. These concepts are the definition of an anisotropic spacing field using a background mesh, the ANNs and the mesh morphing approach for variable geometries. Section 3 contains the main scientific

contribution of this work, including the strategy to identify the anisotropic spacing that is required to generate a mesh suitable to capture a given solution, the strategy to transfer this information to a coarse background mesh and the proposed ANN model. Particular attention is paid to the mathematical properties of the metric tensor to be predicted. In Section 4 two numerical examples are presented. The first example involves the geometry of the ONERA M6 wing with variable free-stream Mach number and angle of attack, whereas the second example considers a full aircraft configuration parametrised with 11 geometric parameters. A study of the suitability of the predicted meshes to perform accurate simulations is also included. Finally, Section 5 summarises the conclusions of the work that has been presented.

2. Background

This section outlines the fundamental concepts necessary to introduce the proposed approach for predicting near-optimal anisotropic spacing, suitable for generating meshes of previously unseen simulations.

2.1. Anisotropic mesh spacing and control using a background mesh

The generation of suitable meshes for simulation often requires a significant level of human expertise and interaction to define a suitable spacing function. Among all the available techniques to define a spacing function, the use of mesh sources and a background mesh provides significant flexibility as it enables the definition of a spacing function that is independent of geometric entities.

This work considers the use of a background mesh for three main reasons. First, previous work concerning the use of ANNs for predicting near-optimal isotropic mesh spacing Lock et al. (2023, 2024); Sanchez-Gamero et al. (2024) has shown that, compared to the prediction of mesh sources, predicting the spacing at a background

mesh leads to higher accuracy for the same amount of available training data. Second, predicting the spacing at a background mesh leads to faster training times given the reduced number of outputs of the ANN, compared to the prediction of a set of mesh sources. Finally, using a background mesh it is easier to define anisotropic spacing functions, which is the main focus on this work, contrary to previous work which was restricted to isotropic spacing.

The process to specify a spacing function starts by creating a background mesh, $\mathcal{B}_h$, defined as a collection of $\mathtt{n_{el}}$ disjoint elements $\{B_e\}_{e=1,\dots,\mathtt{n_{el}}}$ that covers the computational domain Ω, i.e. $\Omega \subseteq \cup_{e=1}^{\mathtt{n_{el}}} B_e$. In this work the background mesh is generated automatically using curvature control to provide local refinement near the region of interest.

When isotropic spacing is of interest, the spacing function is simply defined as a nodal field in the background mesh, namely $\{\delta_i\}_{i=1,\dots,\mathtt{n_{no}}}$, with δ_i the target isotropic spacing at node i, where the number of background mesh nodes is denoted by $\mathtt{n_{no}}$. Once a spacing function is available, the generation of a computational mesh requires the computation of the desired spacing at a point $\boldsymbol{x} \in \Omega$. This operation is performed by identifying the element of the background mesh that contains the point $\boldsymbol{x}$ and using the nodal values of the identified element to interpolate the spacing from the nodal values. In this work the background mesh is always an unstructured tetrahedral mesh so the computation of the spacing at a point uses a linear interpolation of the four nodal values of the identified element.

The definition of an anisotropic spacing function in a background mesh is achieved by associating to each point a metric tensor , which encapsulates the information of three mutually orthogonal directions, $\mathbf{e}_1$, $\mathbf{e}_2$ and $\mathbf{e}_3$, and the desired spacing in each

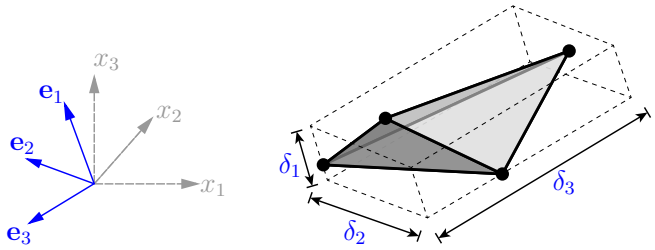


Figure 1: Representation of a tetrahedral element with anisotropic spacing given by three mutually orthogonal directions, $\mathbf{e}_1$, $\mathbf{e}_2$ and $\mathbf{e}_3$, and the desired spacing in each direction, δ_1, δ_2 and δ_3.

direction, δ_1, δ_2 and δ_3. The metric tensor at a point is formally defined as

$$\boldsymbol{\mathcal{M}} = \mathbf{R}\boldsymbol{\Lambda}^2\mathbf{R}^T, \tag{1}$$

where the columns of $\mathbf{R}$ are the vectors $\mathbf{e}_1$, $\mathbf{e}_2$ and $\mathbf{e}_3$ and $\boldsymbol{\Lambda}^2 = \mathrm{diag}\,(1/\delta_1^2, 1/\delta_2^2, 1/\delta_3^2)$. To simplify the presentation it is assumed that the spacings are ordered, i.e. $\delta_1 \leq \delta_2 \leq \delta_3$.

The definition of an element with anisotropic spacing is illustrated in Figure 1.

The generation of an anisotropic mesh requires the definition of a suitable metric tensor at a point in the computational domain, $\boldsymbol{x} \in \Omega$, from the metric tensors defined at the nodes of the element in the background mesh that contains $\boldsymbol{x}$. To this end, a metric interpolation strategy is commonly adopted . In this work, the interpolation of the metric between two points $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ with metrics $\boldsymbol{\mathcal{M}}_1$ and $\boldsymbol{\mathcal{M}}_2$ respectively, is defined as

$$\boldsymbol{\mathcal{M}}(t) = \left((1-t)\,\boldsymbol{\mathcal{M}}_1^{-\frac{1}{2}} + t\,\boldsymbol{\mathcal{M}}_2^{-\frac{1}{2}}\right)^{-2}, \quad \text{for } t \in [0,1]. \tag{2}$$

where the inverse square root of a metric $\boldsymbol{\mathcal{M}}$, as defined in Equation 1 is simply $\boldsymbol{\mathcal{M}}^{-1/2} = \mathbf{R}\boldsymbol{\Lambda}\mathbf{R}^T$, with $\boldsymbol{\Lambda} = \mathrm{diag}\,(\delta_1, \delta_2, \delta_3)$. It is worth noting that this definition ensures a smoother transition of the anisotropy as t varies, when compared to a simple linear interpolation of the metrics, as detailed by Frey and George (2007).

The extension of the metric interpolation for a point $\boldsymbol{x}$ contained in a tetrahedral element is given by

$$\boldsymbol{\mathcal{M}}(t) = \left(\sum_{i=1}^{4} \alpha_i \, \boldsymbol{\mathcal{M}}_i^{-\frac{1}{2}} \right)^{-2} \tag{3}$$

where $\{\alpha_i\}_{i=1,\dots,4}$ are the barycentric coordinates of the point $\boldsymbol{x}$.

2.2. Feed-forward neural networks

A feed-forward artificial neural network (ANN) consists of neurons arranged in layers, where each neuron is connected only to all the neurons in the preceding and succeeding layers. Each connection between two neurons is assigned a weight, and each layer is given a bias, which can be considered an additional neuron with a fixed value. The first layer comprises the ANN inputs, while the neurons in the final layer are referred to as outputs. The intermediate layers, known as hidden layers, are typically indexed as $l = 1, \dots ..., N_l$, where N_l denotes the total number of hidden layers Hagan et al. (1997).

Forward propagation involves computing the value associated with each neuron based on the values of the neurons in the preceding layer, the weights of the corresponding connections, and an activation function. Specifically, the value of the j-th neuron of the $l+1$ layer is computed as

$$z_j^{l+1} = F^{l+1} \left(\sum_{i=1}^{N_n^l} \theta_{ij}^l z_i^l + b_j^l \right), \tag{4}$$

where F^l is the activation function of layer l, θ_{ij}^l is the weight of the connection between the i-th neuron in layer l and the j-th neuron in layer $l+1$, b^l is the bias of layer l, and N_n^l is the number of neurons in layer l. Figure 2 illustrates a schematic representation of a feed-forward ANN.

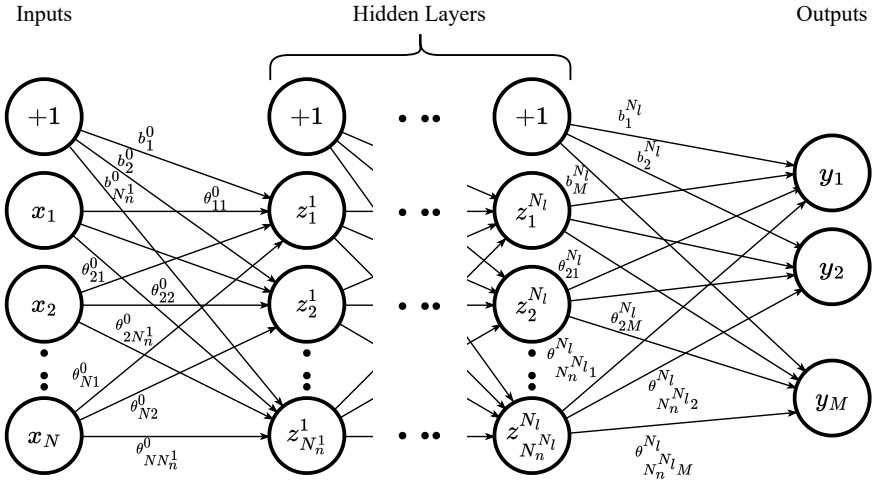


Figure 2: Schematic representation of a feed-forward ANN.

In this work two type of ANNs are considered. One ANN is used to predict the spacing in the associated to the three mutually orthogonal directions that define a metric tensor whereas a different ANN is used to predict the three directions. For the first ANN, the error or cost function is defined as

$$E_\delta(\boldsymbol{\theta}, \mathbf{b}) = \frac{1}{N_{tr} M} \sum_{k=1}^{N_{tr}} \sum_{i=1}^{M} \left[y_i^k(\mathbf{x}^k) - h_i^k(\mathbf{x}^k, \boldsymbol{\theta}, \mathbf{b}) \right]^2, \tag{5}$$

where the output predicted by the ANN is denoted by $h_i^k(\mathbf{x}^k, \boldsymbol{\theta}, \mathbf{b})$, N_{tr} is the number of training cases and the inputs and target outputs are arranged in two vectors $\mathbf{x}^k = \{x_1, ..., x_N\}^T$ and $\mathbf{y}^k = \{y_1, ..., y_M\}^T$, respectively, for $k = 1, \ldots, N_{tr}$.

When the ANN is built to predict the three orthogonal directions, a measure of the alignment between the predicted and target vectors is preferred. To this end, this work proposes the following error function

$$E_e(\boldsymbol{\theta}, \mathbf{b}) = \frac{1}{N_{tr} M} \sum_{k=1}^{N_{tr}} \sum_{i=1}^{M} \left[1 - \mathbf{y}_i^k(\mathbf{x}^k) \cdot \mathbf{h}_i^k(\mathbf{x}^k, \boldsymbol{\theta}, \mathbf{b}) \right]^2, \tag{6}$$

where the target and predicted outputs are now vectors and the deviation of the dot product with respect to one is taken as a measure of alignment between target and predicted directions.

The process of training the ANN consists of finding the weights and biases that minimise the error function. To this end, the ADAM optimiser Kingma and Ba (2014) is employed in this work, with the parameters selected as in Lock et al. (2023, 2024).

As is common in the context of ANNs, the choice of hyperparameters, such as the number of neurons and hidden layers, can significantly impact the accuracy of predictions. In the numerical examples considered, a simple grid search is performed to determine the optimal hyperparameters. To complete the description of the ANN architecture, the activations functions employed need to be detailed. In this work, the sigmoid function is applied in all hidden layers, while a linear function is used in the output layer, namely

$$F^l(z) = \begin{cases} z & \text{if } l = N_l, \\ \dfrac{1}{1+e^{-z}} & \text{otherwise.} \end{cases} \tag{7}$$

2.3. Mesh morphing for geometrically parametrised domains

When considering geometrically parametrised domains, the training cases correspond to available simulations with different geometries. To transfer the spacing function to the same background mesh, as required to train a simple feed-forward ANN, this work proposes the use of a simple mesh morphing approach based on the Delaunay graph .

Given a set of N_g geometric parameters $\boldsymbol{\vartheta} = \{\vartheta_1, \vartheta_2, \ldots, \vartheta_{N_g}\}$ with a pre-defined range for each parameter, $[\vartheta_i^a, \vartheta_i^b]$ for $i = 1, \ldots, N_g$, the background mesh is generated

for the geometric configuration with parameters $\bar{\boldsymbol{\vartheta}} = \{\bar{\vartheta}_1, \bar{\vartheta}_2, \ldots, \bar{\vartheta}_{N_g}\}$, where $\bar{\vartheta}_i = (\vartheta_i^a + \vartheta_i^b)/2$.

A Delaunay tetrahedralisation is built using only the nodes of the background mesh on the boundary of the computational domain. For each mesh node the element of the Delaunay graph that contains that node and its area coordinates are computed and stored.

For a new geometric configuration the boundary mesh nodes of the background mesh are projected to the new geometry and the new position of each mesh node is computed by evaluating the position corresponding to the pre-stored area coordinates in the original configuration.

The proposed strategy enables the morphing of the same background mesh to match the geometry of all the training cases available, ensuring that all cases share the same number of outputs as required to train a simple feed-forward ANN.

3. Near-optimal anisotropic mesh spacing prediction

This section presents a new approach to predict the anisotropic spacing function on a background mesh that can be used to generate meshes suitable for unseen simulations that involve either new geometric configurations or new flow conditions. It is assumed that the data for training is available from historical high-fidelity analysis.

The proposed methodology can be summarised in the following stages:

1. For each solution that is available as training data, identify the anisotropic spacing that could be used to generate a mesh capable of accurately capturing the given solution.
2. For each available training case, transfer the anisotropic spacing to a coarse

background mesh. This step involves the morphing of the background mesh when geometric parameters are considered.

3. Train an ANN to predict the metric tensor that defines the anisotropic spacing at each node of the background mesh.

3.1. Computation of the target anisotropic spacing in the computational mesh

Building on established principles in error analysis Peraire et al. (1992), the determination of a discrete spacing function from a given solution involves computing the Hessian matrix of a selected key variable at each point within the computational mesh. This work considers the generation of meshes for inviscid compressible flow simulations so the selected key variable is the pressure, p.

Pressure is adopted as the key variable for the inviscid transonic external flows considered, since the Hessian of p identifies both the strength and directions of shocks and expansions, which are considered the dominant drivers of anisotropy. In problems where contacts discontinues dominate, other key variables can be considered and multiple key variables can be combined as done, for instance, by Sanchez-Gamero et al. (2024).

Following the equidistribution principle for interpolation error, the spacing at a node $\boldsymbol{x}_i$ in a given direction, defined by a unit vector $\boldsymbol{\beta}$, depends on the derivatives of the pressure as

$$\delta^2_{i,\boldsymbol{\beta}} \left(\sum_{k,l=1}^{\mathtt{n_{sd}}} (H_i)_{kl} \beta_k \beta_l \right) = K, \tag{8}$$

where $\mathtt{n_{sd}}$ denotes the number of spatial dimensions, K is a user-specified constant linked to the level of refinement introduced around areas with high gradient.

The components of the Heassian at a node $\boldsymbol{x}_i$,

$$(H_i)_{kl} = \frac{\partial^2 p_i}{\partial x_k \partial x_l} \tag{9}$$

are evaluated as the derivatives of the derivatives, and the computation of a derivative is performed using the standard finite element recovery postprocess introduced by Zienkiewicz and Zhu (1992a,b) that involves a weighted average of the derivatives in each element of patch associated to a node. A comparison between different approaches to evaluate the derivatives was recently presented by Sanchez-Gamero et al. (2024).

Once these derivatives have been evaluated, the three mutually orthogonal directions at $\boldsymbol{x}_i$ are given by the eigenvectors of the Hessian matrix, $\mathbf{H}_i$, and the spacing in each of those directions is determined as

$$\delta_{i,j} = \begin{cases} \delta_{min} & \text{if } \lambda_{i,j} > K/\delta_{min}^2, \\ \delta_{max} & \text{if } \lambda_{i,j} < K/\delta_{max}^2, \\ \sqrt{K/\lambda_{i,j}} & \text{otherwise,} \end{cases} \tag{10}$$

where $\{\lambda_{i,j}\}_{j=1,\ldots,\mathtt{n_{sd}}}$ are the eigenvalues of $\mathbf{H}_i$.

Equation (10) ensures that the spacings $\delta_{i,j}$ remain within the range $[\delta_{min}, \delta_{max}]$. The lower bound, δ_{min}, prevents excessive local refinement in regions with steep gradients, such as areas near strong shocks. Similarly, the upper bound, δ_{max}, prevents the spacing from becoming too large in regions of undisturbed flow, where the pressure remains smooth or nearly constant. For simplicity in implementation, this study defines δ_{min} and δ_{max} as the minimum and maximum spacing values within the computational mesh, respectively.

The value of K appearing in Equation (10) is defined as

$$K = S^2 \delta_{min}^2 \lambda_{max} \tag{11}$$

where the scaling factor $S \in (0, 1]$ is defined by the user and taken as 0.2 in all the examples. The scaling is introduced to ensure that the refinement is not just

concentrated in one region with a very steep gradient of the key variable, missing most of the other features of the solution. Further details about the effect of altering the value of S are discussed by Sanchez-Gamero et al. (2024).

In addition, in this work, the maximum stretching is restricted to be less than or equal to five. This is based on previous experience that shows that a stretching larger than five leads to an increased number of iterations to converge the solution to steady state with the vertex-centred finite volume solver employed in this work.

3.2. Transfer of the target anisotropic spacing to a background mesh

Given that the different simulations available as training cases might have been performed in different meshes and a feed-forward ANN requires the same number of outputs for all cases, the proposed strategy involves transferring the spacing function of each simulation to the same background mesh. This approach also aims at reducing the number of outputs in the ANN, and consequently the training time, as the available simulations might have been performed in extremely fine meshes.

In Lock et al. (2024); Sanchez-Gamero et al. (2024) the authors proposed a *patchwise-minimum interpolation* approach, also referred to as *conservative interpolation*, to perform this task when the spacing function is isotropic, which is briefly summarised here. Given a node, $\boldsymbol{x}_i^{\mathtt{B}}$, of the background mesh $\mathcal{B}_h$, the patch of elements that contains the node $\boldsymbol{x}_i^{\mathtt{B}}$ is denoted by $\mathcal{P}_i^{\mathtt{B}}$. The subset of nodes of the computational mesh, corresponding to a training case, that are within the patch $\mathcal{P}_i^{\mathtt{B}}$ is denoted by $\mathcal{X}_{\mathcal{P}_i}$. With this notation, the conservative interpolation of the spacing involves defining the spacing at $\boldsymbol{x}_i^{\mathtt{B}}$ as the minimum of the spacing of all nodes in $\mathcal{X}_{\mathcal{P}_i}$, namely

$$\delta_i^{\mathtt{B}} = \min_{j \in \mathcal{X}_{\mathcal{P}_i}} \{\delta_j\}. \tag{12}$$

To transfer the target anisotropic spacing, computed following the strategy described in the previous section, from a computational mesh to a background mesh, this work proposes a *patchwise-minimum metric intersection*. Metric intersection Frey and George (2007) is commonly used to find a representative metric at a point when two (or more) metrics are available. Given two metrics $\boldsymbol{\mathcal{M}}_1$ and $\boldsymbol{\mathcal{M}}_2$, the metric intersection is formally defined as

$$\boldsymbol{\mathcal{M}}_1 \cap \boldsymbol{\mathcal{M}}_2 = \mathbf{P}^{-T} \boldsymbol{\Lambda}_{\max} \mathbf{P}^{-1}, \tag{13}$$

where $\boldsymbol{\Lambda}_{\max} = \operatorname{diag}\left(\max\{\lambda_1, \mu_1\}, \max\{\lambda_2, \mu_2\}, \max\{\lambda_3, \mu_3\}\right)$ with $\lambda_i = \mathbf{e}_i^T \boldsymbol{\mathcal{M}}_1 \mathbf{e}_i$ and $\mu_i = \mathbf{e}_i^T \boldsymbol{\mathcal{M}}_2 \mathbf{e}_i$, for $i = 1, \ldots, \mathtt{n_{sd}}$, and the columns of $\mathbf{P}$ contain the eigenvectors of $\boldsymbol{\mathcal{M}}_1^{-1} \boldsymbol{\mathcal{M}}_2$.

The proposed patchwise-minimum metric intersection to transfer an anisotropic spacing from a computational mesh to a background mesh is described next. Given a node, $\boldsymbol{x}_i^{\mathtt{B}}$, of the background mesh $\mathcal{B}_h$, the patch of elements that contains the node $\boldsymbol{x}_i^{\mathtt{B}}$, denoted by $\mathcal{P}_i^{\mathtt{B}}$, is considered. The subset of nodes of the computational mesh that are within the patch $\mathcal{P}_i^{\mathtt{B}}$ is denoted by $\mathcal{X}_{\mathcal{P}_i}$. With this notation, the patchwise-minimum metric intersection involves defining the metric for node $\boldsymbol{x}_i^{\mathtt{B}}$ as

$$\boldsymbol{\mathcal{M}}_i^{\mathtt{B}} = \bigcap_{j \in \mathcal{X}_{\mathcal{P}_i}} \boldsymbol{\mathcal{M}}_j. \tag{14}$$

Remark 1. *As reported in Sanchez-Gamero et al. (2024), special attention must be paid to two special cases. First, if the patch of a background mesh node, $\mathcal{P}_i^{\mathcal{B}}$, does not contain any node of the computational mesh, i.e $\mathcal{P}_i^{\mathcal{B}} = \emptyset$, then the proposed strategy is to utilise the metrics of the nodes of the computational mesh that contain the current background mesh node. Second, for domains with curved boundaries, some nodes of the computational mesh might not belong to any patch associated to the background mesh and, consequently, the approach described above would ignore*

the metrics defined at those points. To avoid this, the nodes of the computational mesh that do not belong to any element of the background mesh are associated to the closest elements of the background mesh. This ensures that all the metrics available in the computational mesh are utilised when transferring information to the background mesh.

3.3. ANN architecture to predict anisotropic spacing

Using the strategies described in the two previous sections, the tensor metric that defines the target anisotropic spacing in a common background mesh is available, for a set of training cases.

A potential ANN architecture would involve setting as inputs of the ANN the desired parameters (e.g. flow conditions or geometric parameters) and defining as outputs the nine components of the metric tensor at all the nodes of the background mesh. However, such a strategy would ignore that a metric tensor must be given by a symmetric definite positive matrix. In addition, predicting the components of the metric tensor directly lacks interpretability of the predictions, as the information about the three mutually orthogonal directions and the spacing in each direction cannot be easily observed by looking at the components of the metric tensor directly.

The orthogonality property of the directions to be predicted can be exploited to reduce the amount of information to be predicted and at the same time ensure that the resulting metric tensor satisfies the required properties, i.e. it is given by a symmetric definite positive matrix. To this end, the first direction (corresponding to the minimum spacing) is expressed using spherical coordinates as, meaning that only two angles are required, namely α_1 and α_2. Next, to strongly enforce the required orthogonality, the second direction must lie within the orthogonal plane to the first direction, meaning that it can be expressed in polar coordinates in the orthogonal

plane using a single angle, namely α_3. Finally, the third direction is uniquely determined from the orthogonality property and there is no need to characterise or predict this direction.

The in-plane reference for α_3 is obtained by first "undoing" the orientation of the first direction. More precisely, using the spherical angles of $\mathbf{e}_1$ (α_1 and α_2) to rotate the global frame so that $\mathbf{e}_1$ would align with the vertical axis; applying this inverse rotation to the fixed vector $-\mathbf{e}_y$ yields a datum vector $\mathbf{r}_0$ that lies in the plane orthogonal to $\mathbf{e}_1$. The angle α_3 is then defined as the angle between $\mathbf{r}_0$ and the second direction $\mathbf{e}_2$ within that plane. As the metric is invariant to $\mathbf{e}_2 \to -\mathbf{e}_2$, only the magnitude in $[0, \pi]$ is needed. In practice, when $\mathbf{e}_1$ is nearly parallel to $\mathbf{e}_y$, a perturbed reference is used in place of $-\mathbf{e}_y$ to avoid a degenerate projection.

Remark 2. *Given the definition of a metric tensor in Equation* (1)*, it is easy to verify that the metric tensor given by three spacial directions* $\gamma\mathbf{e}_1$*,* $\mathbf{e}_2$ *and* $\mathbf{e}_3$*, with spacings* δ_1*,* δ_2 *and* δ_3*, respectively, is identical if* $\gamma = 1$ *or* $\gamma = -1$*. Obviously, the same applies if the change of sign is applied to any of the three directions.*

The observation in Remark 2 implies that the angles can be restricted to $\alpha_1 \in [0, \pi]$, $\alpha_2 \in [-\pi/2, \pi/2]$ and $\alpha_3 \in [0, \pi]$. This observation also implies that two dissimilar angles, for instance ϵ and $\pi - \epsilon$, for a small angle ϵ, would produce almost identical metric tensors. It is therefore not advisable to define the angles as the outputs of the ANN as a small variation of one anisotropic variation can lead to a large variation of the angle, making the training more difficult and leading to a lower prediction accuracy.

To avoid this issue, each angle is independently mapped to an imaginary circle in the two dimensional plane, namely $\boldsymbol{v}_i = (\sin(\alpha_i), \cos(\alpha_i))$, ensuring that two dissimilar angles corresponding to almost identical orientations lead to almost identical

vectors $\boldsymbol{v}_i$.

Remark 3. *Two degenerate cases can appear in the proposed framework. First, when the spacing is nearly isotropic ($\delta_1 \approx \delta_2 \approx \delta_3$), the principal directions are indeterminate. Second, in near-transverse anisotropy, where two spacings are nearly equal, α_3 becomes uninformative. These situations might be seen as introducing noise during training but in both cases the predicted angles are not relevant so this has no effect on the accuracy of the predictions in these degenerate cases.*

Therefore, the proposed approach consists of building an ANN where the inputs are flow conditions or geometric parameters and the outputs are the unit vectors $\boldsymbol{v}_i$ and the corresponding spacings δ_i for $i = 1, \ldots, \mathtt{n_{sd}}$.

Three models have been developed and compared in the numerical examples shown later. The first model consists of training two different ANNs, one ANN to predict the three spacings and a second ANN to predict the three vectors $\boldsymbol{v}_i$ that uniquely define the first two directions of anisotropy. The second model consists of training three different ANNs, one ANN to predict the three spacings, a second ANN to predict the two vectors $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ that uniquely define the first direction of anisotropy and a third ANN to predict the vector $\boldsymbol{v}_3$ that uniquely defines the second direction of anisotropy. The third model investigated involves training four different ANNs, one ANN to predict the three spacings and three more ANNs to independently predict the vectors $\boldsymbol{v}_1$, $\boldsymbol{v}_2$ and $\boldsymbol{v}_3$. Table 1 summarises the three models described.

In terms of implementation, TensorFlow 2.7.0 Abadi et al. (2016) is used to construct and train the ANNs. To minimise the effect of the random initialisation of the ANN weights, each training experiment is repeated five times with different random seeds. In all the examples, each training is performed for a maximum of

Table 1: Three models used for training ANNs to predict the anisotropic spacing on the background mesh.

Model	ANN architecture			
1	ANN_1 (δ_1, δ_2, δ_3)	ANN_2 (α_1, α_2, α_3)		
2	ANN_1 (δ_1, δ_2, δ_3)	ANN_2 (α_1, α_2)	ANN_3 (α_3)	
3	ANN_1 (δ_1, δ_2, δ_3)	ANN_2 (α_1)	ANN_3 (α_2)	ANN_4 (α_3)

5,000 epochs, with early stopping applied if no improvement in the objective function is observed for 100 consecutive epochs. The batch size used in all experiments is eight, which produced better performance compared to the default value of 32 in TensorFlow.

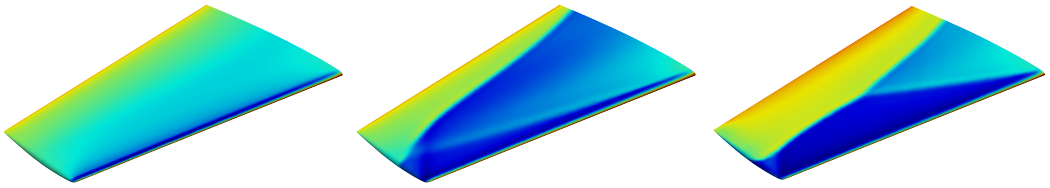
Finally, to evaluate the prediction accuracy of the trained ANNs, model performance is measured using the mean absolute error (MAE), given by

$$\text{MAE} = \frac{1}{M}\sum_{i=1}^{M}|\hat{y}_i - h_i|.$$

where y_i and h_n denote the target and ANN predicted values, respectively. This quantity is reported per output, as it provides an absolute, unit-consistent measure, that is easy to interpret. However, as discussed in Remark 3, the error in direction for elements that are nearly isotropic is not indicative of mesh quality, as orientation is irrelevant when the metric is nearly isotropic.

4. Numerical examples

This section presents two numerical examples to assess the accuracy of the proposed strategy to predict the anisotropic spacing for unseen simulations. The first example involves a problem with a fixed wing geometry and variable flow conditions



(a) $M_\infty = 0.660, \alpha = 1.91°$ (b) $M_\infty = 0.885, \alpha = 2.13°$ (c) $M_\infty = 0.809, \alpha = 7.12°$

Figure 3: ONERA M6 wing: Pressure coefficient, C_p, for three different flow conditions.

characterised by two parameters. The second example involves a more complex problem with fixed flow conditions and 11 geometric parameters that characterise the shape of a wing in a full aircraft configuration.

All the data used in the examples corresponds to inviscid flow simulations performed with the FLITE system Sørensen et al. (2003), a well established vertex-centred finite volume solver.

4.1. Anisotropic spacing predictions on the ONERA M6 wing at various inflow conditions

The first example considers the inviscid compressible flow past the ONERA M6 wing for varying flow conditions, characterised by the free-stream Mach number, M_∞, and the angle of attack α. The variation of the flow conditions is $M_\infty \in [0.6, 0.95]$ and $\alpha \in [0°, 8°]$, encompassing subsonic and transonic flow regimes and leading to a substantial variation of the spacing function required for each simulation.

Figure 3 shows three different pressure coefficient distributions over the wing, corresponding to different flow conditions. The first case corresponds to a subsonic flow where the gradients are mainly concentrated along the leading and trailing edges and anisotropic spacing would be beneficial in these regions. The other two cases involve a transonic flow with a λ-shape shock, but with significant variability of the

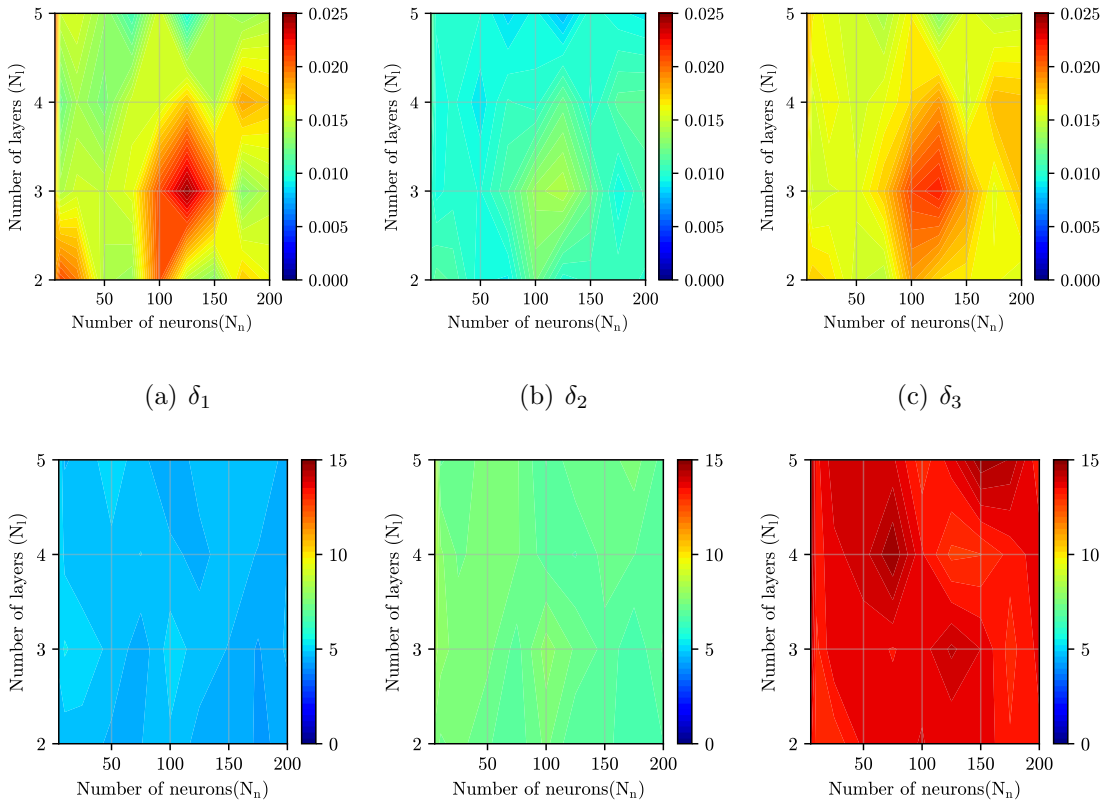
regions containing steep gradients.

All the simulations available, including training and test cases, were computed on the same tetrahedral mesh with 4.6M elements and 782K nodes and using isotropic spacing. To select the training and test cases, sampling is performed using Halton sequencing Halton (1964). More precisely, scrambled Halton sequencing is employed as it is known to maintain the low-discrepancy in high-dimensional problems Vandewoestyne and Cools (2006).

Remark 4. *When utilising historical data available in industry, the datasets are unlikely to align with a Halton sequencing. Instead, it is anticipated that the sampling, done by an expert engineer, would be denser in critical regions of the flight envelope where significant changes in flow features are expected to occur. Therefore, although the influence of the sampling method is out of the scope of the current work, it is anticipated that a biased sampling, as done by an expert engineer, would lead to better performance of the trained ANN for the same amount of training data or to similar performance with less training data.*

For each available case, the metric tensor is computed at each node of the computational mesh using the procedure described in Section 3.1. Next, employing the strategy presented in Section 3.2, for each training case, the metric is transferred to the same background mesh, which in this example has approximately 590K elements and 100K nodes. Finally, using the procedure described in Section 3.3, the spacing and the vectors that describe the first two anisotropic directions are computed, for each case and for each node of the background mesh.

With this information ANNs are trained using the parameters described in Section 3.3. For the first model, described in Table 1, where two ANNs are trained separately to predict spacing and anisotropy directions, Figure 4 shows the mean



(a) δ_1 (b) δ_2 (c) δ_3

(d) α_1 (e) α_2 (f) α_3

Figure 4: ONERA M6 wing: MAE for the spacings (δ_i) and angles (α_i) as a function of the number of layers and number of neurons in each layer employing the first model of Table 1.

average error (MAE) for ANNs with an increasing number of hidden layers and neurons, when using $N_{tr} = 40$. The results clearly show that predicting the spacing is significantly easier than predicting the anisotropic directions accurately, with the MAE for spacings being two or three orders of magnitude lower than that for the directions. The number of hidden layers utilised varies from two to five because previous studies involving the prediction of isotropic spacing Lock et al. (2024) showed that one hidden layer produces significantly less accurate results and six layers do

not bring any benefit in terms of accuracy. In fact, as shown in Figure 4 using more than two layers does not provide any extra accuracy. The results also show that the most accurate results are obtained for the first anisotropic direction and its associated spacing. Given that the first direction is associated to the smaller spacing, this is the most important direction to be accurately predicted in order to capture the steep gradients near shocks.

The number of neurons is varied between five and 200, but the results show that using more than 50 neurons does not provide any significant benefit. This means that the training times of the ANN are low. For instance using $N_{tr} = 40$ two hidden layers and five neurons per layer, the training of the ANN to predict spacing with takes approximately seven minutes, whereas the ANN to predict the anisotropic directions takes 35 minutes. When the number of neurons is increased to 100 per layer, the training of the ANN to predict spacing with takes approximately 23 minutes, whereas the ANN to predict the anisotropic directions takes 67 minutes.

Next, the effect of the size of the training dataset is studied. For the first model described in Table 1, where two independent ANNs are trained, Figure 5 shows the maximum MAE for all test cases as a function of the number of training cases for the six predicted outputs. The ANN architecture used in this study considers the hyperparameters from the best performing ANN, as identified in the hyperparameter tuning process shown in Figure 4.

Figure 5 shows a consistent decay of MAE with the number of training cases across all six outputs. The most pronounced improvement occurs between 20 and 40 cases: the curves steepen in this range, indicating that the additional data is particularly informative. Beyond $N_{tr} \approx 40$ the spacing errors δ_i continue to decrease but with clearly diminishing returns, while the angle errors α_i also improve yet with a gentler slope. Among the angles, α_3 remains the most challenging but still bene-

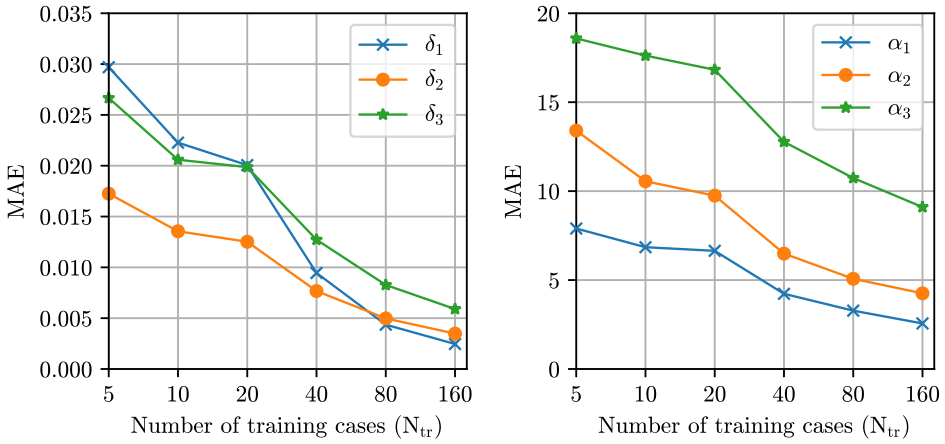
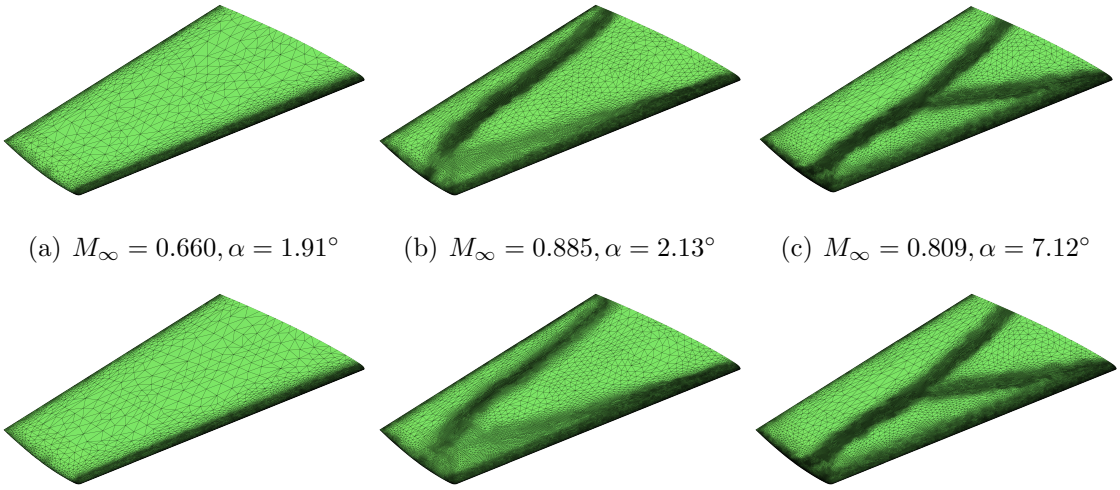


Figure 5: ONERA M6 wing: MAE for the spacings (δ_i) and angles (α_i) as a function of the number of training cases, N_{tr}, employing the first model of Table 1.

fits steadily from more data. Taken together, these trends indicate a clear elbow at about 40 training cases: this setting captures the bulk of the achievable reduction for the spacings and a substantial share for the angles, with further increases yielding incremental gains. We therefore regard $N_{tr} = 40$ as a practical minimum that balances accuracy and training cost for the present application.

To visually illustrate the potential of the proposed approach, the trained ANNs for the first model of Table 1 are next used to predict the anisotropic spacing field, which is subsequently used to generate an anisotropic mesh. Figure 6 shows the target mesh and the predictions for three inflow conditions corresponding to test cases, unseen by the ANN during training. The target meshes are generated by directly using the metric defined in the background mesh. The predicted meshes closely follow the refinement patterns of the target meshes, effectively capturing key features of the flow. For the first case, which is a subsonic test case, the refinement is concentrated in the leading and trailing edges and the anisotropy of the predicted



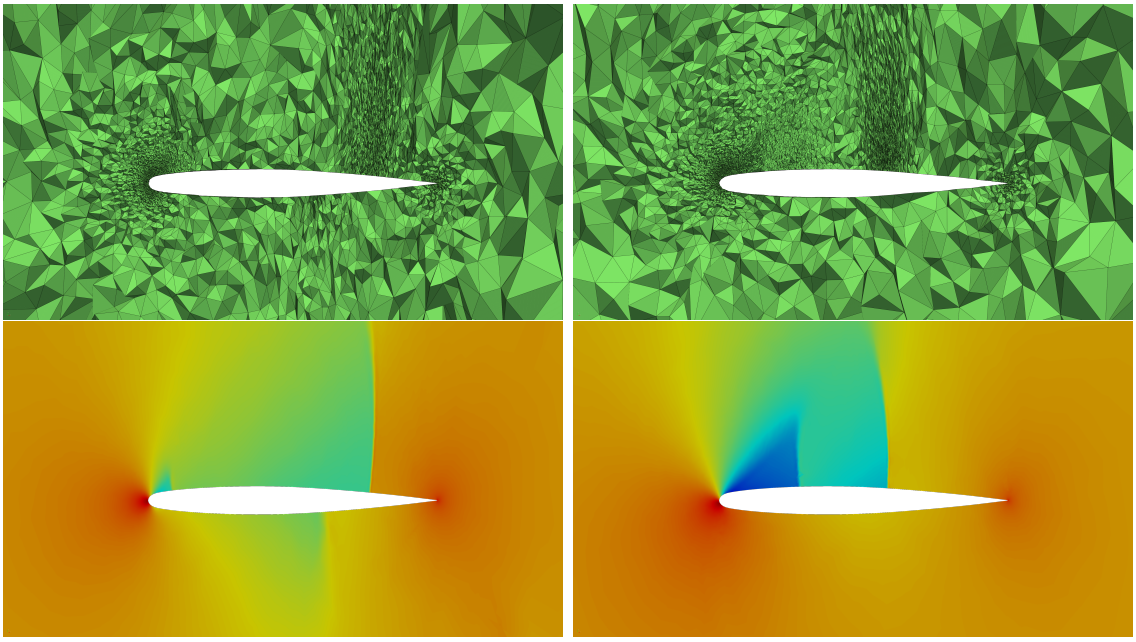
(a) $M_\infty = 0.660, \alpha = 1.91°$ (b) $M_\infty = 0.885, \alpha = 2.13°$ (c) $M_\infty = 0.809, \alpha = 7.12°$

(d) $M_\infty = 0.660, \alpha = 1.91°$ (e) $M_\infty = 0.885, \alpha = 2.13°$ (f) $M_\infty = 0.809, \alpha = 7.12°$

Figure 6: ONERA M6 wing: Target (top) and predicted (bottom) meshes using the first model for three flow conditions unseen by the ANN during training.

spacing matches the one observed in the target mesh. For the two transonic cases, the predicted meshes provide the refinement required to capture the shocks and, again, the anisotropic character of the target spacing is clearly observed in the predicted meshes. The anisotropic spacing is better illustrated for the two transonic cases by showing the volume mesh together with the corresponding C_p solution in Figure 7. For both cases, the element anisotropy aligns closely with the shock captured in the solution, with elements elongated tangentially to the shock and refined in the normal direction.

To quantify the accuracy of the 3D volume mesh predictions and to compare the different models in Table 1, the spacing function predicted by the ANN is compared with the target spacing function. It is worth noting that the spacings cannot be directly compared as the target and predicted directions of anisotropy are not identical. Therefore, to produce a suitable error measure, the predicted anisotropic

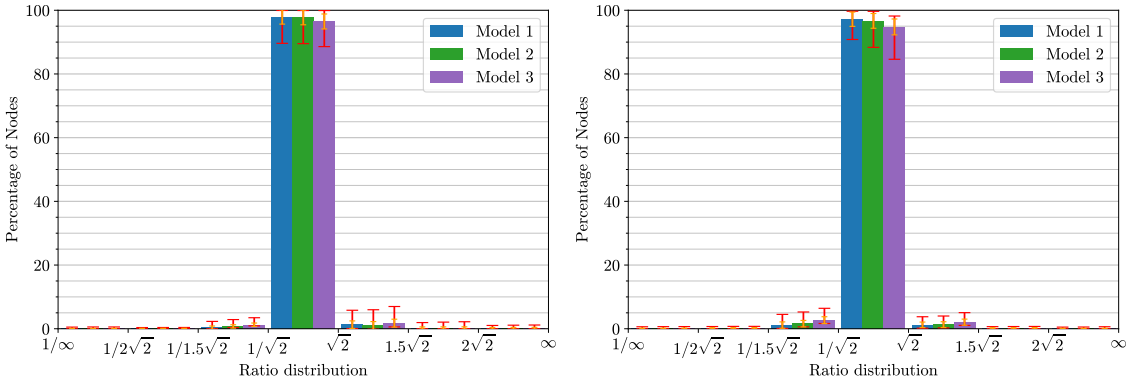


(a) $M_\infty = 0.885, \alpha = 2.13°$ (b) $M_\infty = 0.809, \alpha = 7.12°$

Figure 7: ONERA M6 wing: predicted volume meshes (top) and corresponding pressure coefficient distributions (bottom) for two transonic flow conditions unseen during training. Both views show the same clipped plane at $y = 5$, with the lower plots displaying the C_P field computed using the corresponding mesh above.

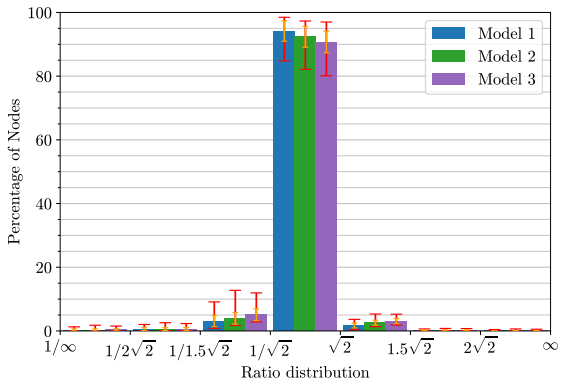
spacings are projected onto the target spacing directions given by $\mathbf{e}_1$, $\mathbf{e}_2$ and $\mathbf{e}_3$ for each mesh node.

Figure 8 presents histograms comparing the three models of Table 1, evaluated with $N_{tr} = 40$. The histograms summarise, for each direction $\mathbf{e}_1$, $\mathbf{e}_2$, and $\mathbf{e}_3$, how the predicted spacing compares with the target across all test cases. For every bin, we first compute—case by case—the percentage of nodes that fall in that bin; the bar height is the average of those percentages over all cases. The red error bars show the full range across test cases, and the orange markers show the typical spread around the average (one standard deviation). Bins are placed at ratios that are multiples



(a) Spacing along $\mathbf{e}_1$

(b) Spacing along $\mathbf{e}_2$



(c) Spacing along $\mathbf{e}_3$

Figure 8: ONERA M6 wing: Histogram of the ratio between the predicted and target spacings for the three models of Table 1.

of the square root of two and its inverse, reflecting the standard convention that spacings within a factor of $\sqrt{2}$ of the target are acceptable. Ratios below $1/\sqrt{2}$ indicate the mesh is finer than required, whereas ratios above $\sqrt{2}$ indicate that the mesh is not refined enough.

For each axis $\mathbf{e}_j$ and each test case $n = 1, \ldots, N_{\text{test}}$, we build a histogram of the spacing ratio $r = \hat{h}_j / h_j^\star$ using the bins shown; bin counts are normalised to percentages so they sum to 100% per case. The bar height is the mean percentage over

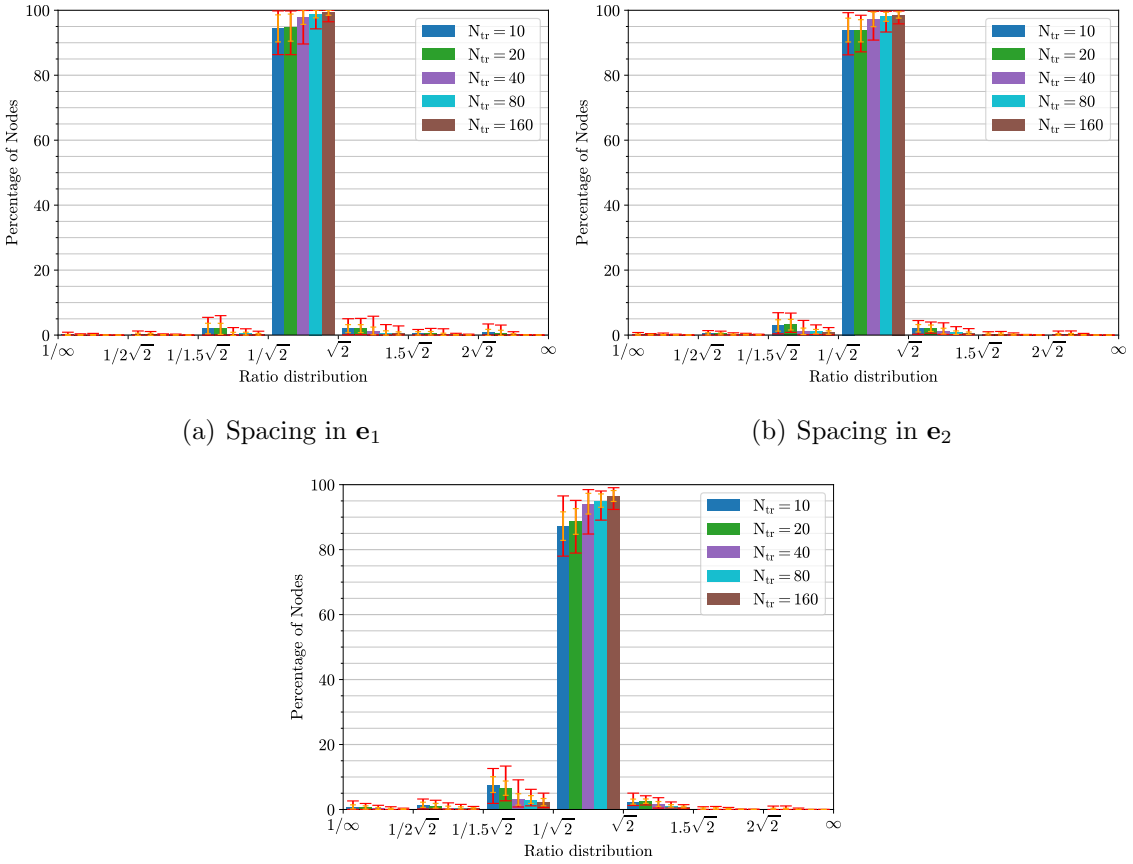
all N_{test} cases. The red error bars show, for each bin, the minimum and maximum percentage observed across the N_{test} cases (range). The orange error bars indicate one standard deviation about the mean across cases for that bin. This conveys both the central tendency and the inter-case variability of the spacing-ratio distribution.

The results show that model 1, with two ANNs to predict spacing and directions separately, leads to the most accurate predictions for all directions of the target anisotropic spacing. Slightly less accurate results are obtained for the second model, where the angles associated to the first and second anisotropic directions are predicted by two independent ANNs. Finally, the least accurate results are observed for the third model, where each angle is predicted by a different ANN. This indicates that using a single ANN to predict all the information about anisotropic directions is not only more efficient but also provides the most accurate predictions.

Very small differences between the accuracy of the three models are observed on the predicted spacing along the first target anisotropic direction, $\mathbf{e}_1$, with almost 95% of the mesh nodes having an acceptable prediction with the first model. For the second target anisotropic direction, $\mathbf{e}_2$, the third model shows a sizeable loss of accuracy, whereas the first and second models have almost the same accuracy. The better performance of the first model is best appreciated when considering the spacing along the third target anisotropic direction, $\mathbf{e}_3$. It is also worth mentioning that the first model is the one that exhibits the minimum areas of under-refinement.

The best performance of the first model is attributed to its ability to leverage global information during training as it is the only model that predicts the two first directions of anisotropy together.

To further quantify the accuracy of the predictions in terms of the training dataset size, Figure 9 shows the histogram of the anisotropic spacing accuracy for the first model in terms of the number of training cases, N_{tr}. The histograms and error



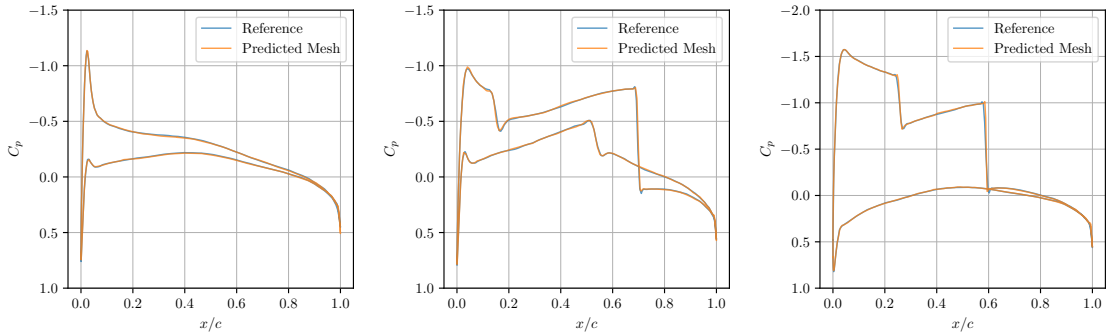
(a) Spacing in $\mathbf{e}_1$

(b) Spacing in $\mathbf{e}_2$

(c) Spacing in $\mathbf{e}_3$

Figure 9: ONERA M6 wing: Histogram of the ratio between the predicted and target spacings for the first model of Table 1 and for an increasing number of training cases, N_{tr}.

bars are constructed identically to Figure 8. The histograms show a significant improvement in prediction accuracy for $N_{tr} \geq 40$, as previously shown with the MAE graphs in Figure 5. It is worth noting that, for $N_{tr} = 40$ the percentage of nodes with a prediction of the full metric tensor reaches almost 95%. Furthermore, with only 20 training cases, the prediction of the spacing in the first direction (the most critical to accurately represent the shocks) is acceptable for almost 95% of the



(a) $M_\infty = 0.660, \alpha = 1.91°$ (b) $M_\infty = 0.885, \alpha = 2.13°$ (c) $M_\infty = 0.809, \alpha = 7.12°$

Figure 10: ONERA M6 wing: Comparison of the pressure coefficient, C_p, for three different unseen flow conditions, at one section.

nodes.

To conclude this example, the suitability of the meshes predicted to perform simulations for unseen cases is studied. To this end, an anisotropic spacing is predicted for three unseen cases, the corresponding anisotropic meshes are generated and simulations are performed. The results, involving pressure coefficient distribution and the lift and drag coefficients, are compared to the results obtained with a fine reference mesh which is produced using mesh adaptivity, starting with the original isotropic mesh with 4.6M elements used to generate all the datasets.

Figure 10 compares the pressure coefficient distributions at one wing section for three flow conditions unseen during ANN training. The plot shows excellent agreement between the two, with the predicted–mesh and reference–mesh curves being nearly coincident along the section.

To confirm the suitability of the predicted meshes, Table 2 compares the reference lift and drag coefficients, C_L and C_D respectively, with the one obtained after performing a simulation with the predicted meshes. The results show a maximum

Table 2: ONERA M6 wing: Reference aerodynamic coefficients and values computed with the predicted meshes.

	$M_\infty = 0.660, \alpha = 1.91°$		$M_\infty = 0.885, \alpha = 2.13°$		$M_\infty = 0.809, \alpha = 7.12°$	
	Reference	Predicted	Reference	Predicted	Reference	Predicted
C_L	0.146	0.147	0.240	0.241	0.645	0.646
C_D	0.0014	0.0013	0.0142	0.0143	0.0628	0.0630

variation in the lift and drag coefficients of only one lift count ($\Delta C_L = 10^{-3}$) and two drag counts ($\Delta C_D = 2 \times 10^{-4}$), respectively, clearly demonstrating the suitability of the predicted meshes to perform simulations of unseen cases.

4.2. Anisotropic spacing predictions on a geometrically parametrised aircraft

The second example considers the anisotropic spacing prediction for a full aircraft configuration that is geometrically parametrised at fixed transonic flow conditions corresponding to free-stream Mach number $M_\infty = 0.78$ and angle of attack $\alpha = 2.0°$.

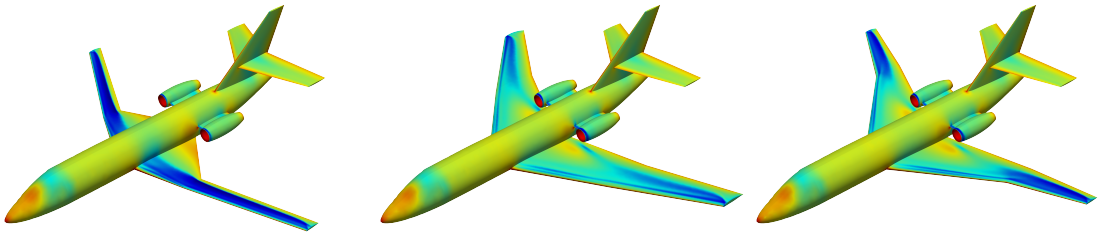
The wings of the aircraft are parametrised using 11 geometric parameters, with the details given in Table 3 and for a fixed total semi-span of 8.15m.

To illustrate the variability in the geometry and the corresponding solution induced by the selected geometric parameters, Figure 11 depicts the pressure coefficient for three different geometric configurations. In the three cases shown, a strong shock wave is observed on the upper surface of the wing, clearly displaying the transonic nature of the flow.

All three cases exhibit a strong transonic shock on the upper wing surface. In Geometry 1 the shock is nearly spanwise-uniform over most of the wing. In Geometry 2 a localised pressure disturbance (a compression–expansion pattern) is visible on the

Table 3: Geometrically parametrised aircraft: design parameters and the range of variation for each parameter.

Parameter	**Lower Limit**	**Upper Limit**
Inboard sweep	15°	50°
Outboard sweep	15°	60°
Inboard semi-span	2.0 m	5.0 m
Inboard dihedral	−7.5°	7.5°
Outboard dihedral	−7.5°	7.5°
Mid-span twist	0°	3°
Tip twist	−2°	3°
Mid-span chord ratio	0.25	0.75
Tip chord ratio	0.25	0.75
Mid-span thickness ratio	0.25	0.75
Tip thickness ratio	0.25	0.75



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

Figure 11: Geometrically parametrised aircraft: Pressure coefficient, C_p, for three different geometric configurations.

upper wing immediately ahead of the nacelle/intake, and a slight inboard kink of the shock is observed. In Geometry 3 the outboard shock appears stronger and more

swept, with a similar disturbance ahead of the nacelle and a visible kink near the pylon area. We simply note these co-located features; establishing causality is beyond the scope of this figure. While the differences are subtle at this colour scale, they indicate geometry-dependent variations in shock position and strength, underscoring the need for meshes tailored to each configuration.

As in the previous example, the strategy presented in the previous section is used to compute the metric tensor for each available simulation and to transfer the spacing to the background mesh, which in this example has approximately 847K elements and 143K nodes.

The ANNs are trained using the same parameters as in the previous example, as described in Section 3.3. Given the better performance and efficiency of the first model described in Table 1, this is the only model considered in this example, where two ANNs are trained separately to predict spacing and anisotropy directions. Figure 12 shows the MAE for ANNs with an increasing number of hidden layers and neurons, when using $N_{tr} = 40$. The results show a qualitative behaviour similar to the previous example despite the different nature of the parameters and the increased dimensionality of the problem. The prediction of the spacing is again significantly more accurate than the prediction of the angles that define the directions of anisotropy. The main difference with respect to the previous example is that the prediction of the angles that define the anisotropic direction is more challenging, in particular the second angle, α_2. This is expected given the fact that the same amount of data is utilised, $N_{tr} = 40$, but this problem has 11 geometric parameters.

To study the effect of the training dataset size, Figure 13 the maximum MAE for all test cases as a function of the number of training cases for the six predicted outputs. As in the previous example the error decreases as the number of training cases increases. Despite the increased difficulty of this problem, it is remarkable to

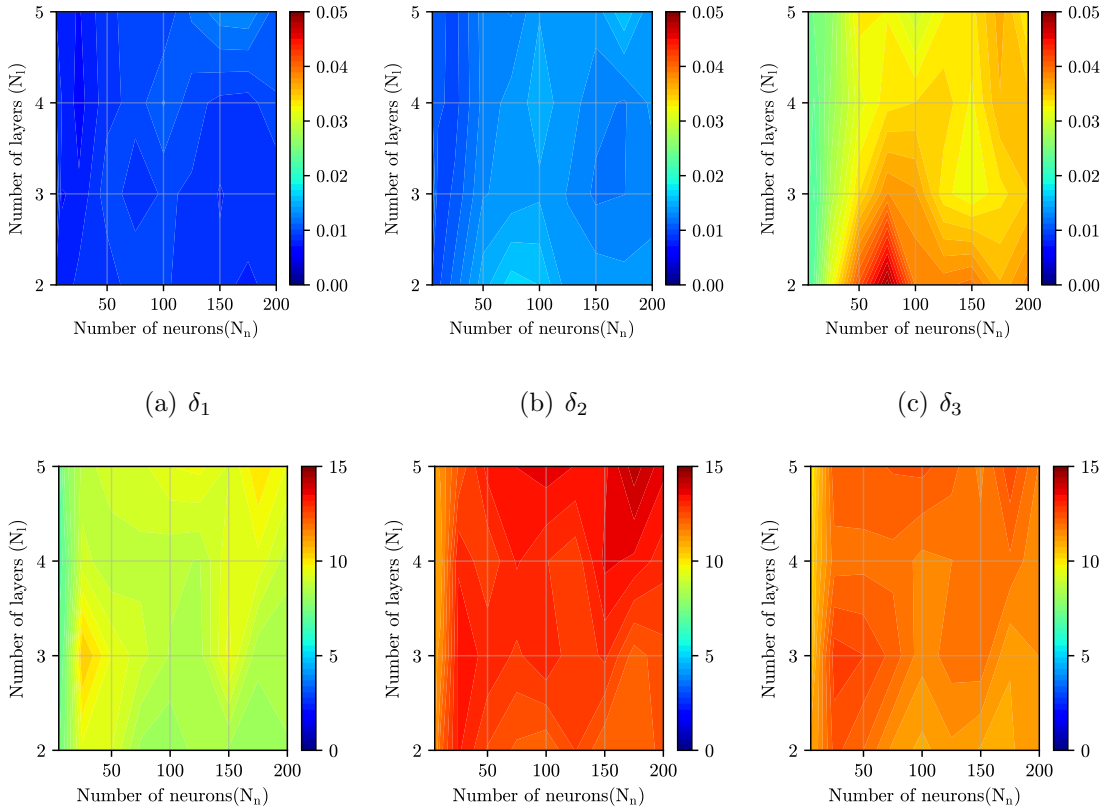


Figure 12: Geometrically parametrised aircraft: MAE for the spacings (δ_i) and angles (α_i) as a function of the number of layers and number of neurons in each layer employing the first model of Table 1.

observe a significant gain in accuracy with only 40 training cases.

To illustrate the ability of the proposed approach to predict anisotropic spacing for a more complex problem, the trained ANNs are used to predict the metric tensor at each point of the domain, which is subsequently used to generate an anisotropic mesh for three cases unseen during training. Figure 14 compares the target and predicted meshes for the three cases. The results clearly show the ability of the

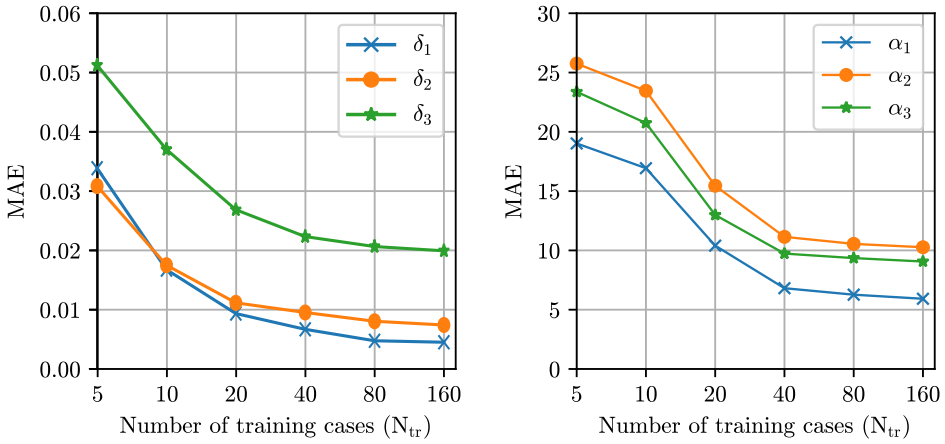
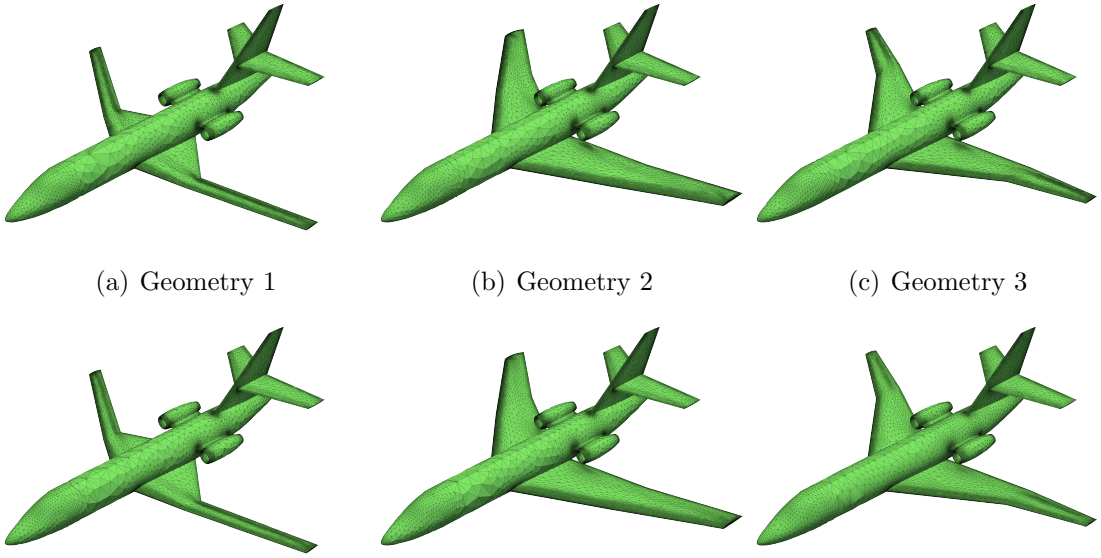


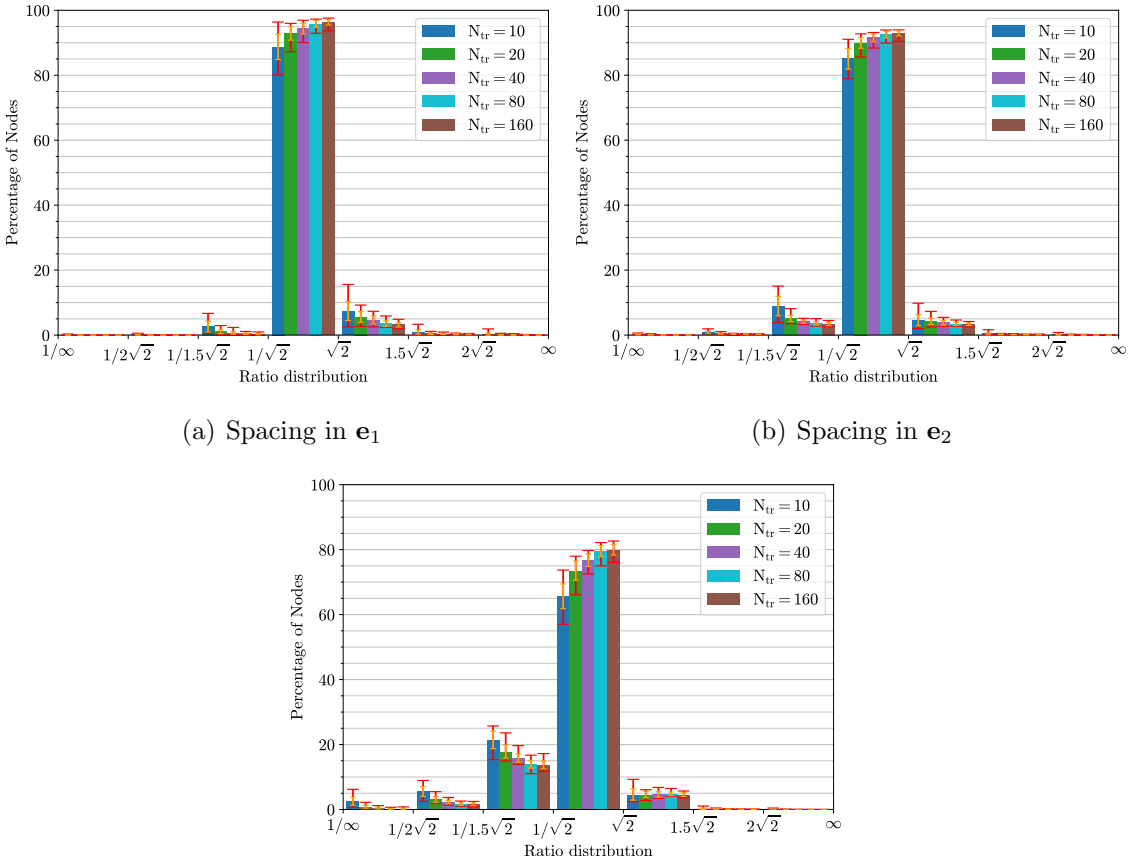
Figure 13: Geometrically parametrised aircraft: MAE for the spacings (δ_i) and angles (α_i) as a function of the number of training cases, N_{tr}, employing the first model of Table 1.



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

(d) Geometry 1 (e) Geometry 2 (f) Geometry 3

Figure 14: Geometrically parametrised aircraft: Target (top) and predicted (bottom) meshes for three geometric configurations unseen by the ANN during training.



(a) Spacing in $\mathbf{e}_1$

(b) Spacing in $\mathbf{e}_2$

(c) Spacing in $\mathbf{e}_3$

Figure 15: Geometrically parametrised aircraft: Histogram of the ratio between the predicted and target spacings for an increasing number of training cases, N_{tr}.

trained ANN to accurately predict the regions where refinement is required as well as the anisotropic character of the target spacing.

A more quantitative analysis is provided in Figure 15, showing the histogram of the anisotropic spacing accuracy for different training datasets. The histograms show that with only 20 training cases it is possible to predict 90% of the nodes with an acceptable spacing in the two first directions. It is remarkable that the accuracy in

the first two directions is similar to the one obtained in the previous example, despite in this scenario 11 geometric parameters are considered and in the previous example only two flow conditions were used. The main difference between the two examples is that in the current example it is particularly challenging to predict the spacing in the third direction. However, it is worth noting that this direction corresponds to the maximum of the three spacings and therefore the least relevant. Furthermore, the histograms show that the trained ANN tends to refine more than required in this direction rather than under-refine, which is obviously preferable.

5. Concluding remarks

A methodology to predict the near-optimal anisotropic spacing function for unseen simulations has been presented. The strategy aims to leverage the vast amount of high-fidelity data that is normally available in industry to build an ANN capable of predicting anisotropic spacing.

The strategy involves computing the metric tensor at each mesh node and for each available simulation and transferring this information to a common coarse background mesh. The strategy to transfer the anisotropic spacing to the background meshes uses a patchwise-minimum interpolation using metric intersection approach aimed at minimising the possibility of producing meshes that miss key solution features.

Two ANN are trained independently to predict the directions of anisotropy and the corresponding spacings. The proposed ANN architecture exploits the mathematical properties of a metric tensor and predicts only two directions of anisotropy and three spacings. Different models were tested and compared to investigate if training different ANNs to predict the angles that define the first two anisotropic directions separately were beneficial. It was found that predicting the angles that define the

two first directions of anisotropy using a single ANN provided the most accurate predictions.

Numerical examples involving three-dimensional inviscid compressible flow simulations are used to illustrate the potential of the proposed strategy and the accuracy of the predicted spacing functions. For an example with a fixed geometry and two parameters characterising the flow conditions, it was found that with only 10 training cases the ANN can predict an acceptable spacing in the first direction of anisotropy for almost 95% of the nodes. It is worth remarking that the first direction of anisotropy is the most critical as it corresponds to the direction of minimum spacing. The predicted meshes were utilised to perform simulations and confirm that acceptable results can be obtained in terms quantities of interest such as lift and drag.

The second example, involving a full aircraft with 11 geometric parameters, was used to demonstrate the applicability in a more complex industrial setting. Despite the higher dimensionality and more complex flow features, the results showed that with 40 training cases it was possible to predict an acceptable spacing in the first direction of anisotropy for almost 95% of the nodes.

Future work will focus on the possibility of using transfer learning to train new ANNs for new problems to minimise the amount of training data to be generated and to avoid training ANNs from scratch. This strategy has the potential to provide a tool that can be deployed in an industrial setting.

Acknowledgements

The authors are grateful for the financial support provided by the Engineering and Physical Sciences Research Council (EP/T517987/1).

Declaration of interest

The authors have no competing interests to declare that are relevant to the content of this article.

References

W. Dawes, P. Dhanasekaran, A. Demargne, W. Kellar, A. Savill, Reducing bottlenecks in the CAD-to-mesh-to-solution cycle time to allow CFD to participate in design, Journal of Turbomachinery 123 (2001) 552–557.

J. P. Slotnick, A. Khodadoust, J. Alonso, D. Darmofal, W. Gropp, E. Lurie, D. J. Mavriplis, CFD vision 2030 study: a path to revolutionary computational aerosciences, Technical Report, 2014.

S. L. Karman, N. Wyman, J. P. Steinbrenner, Mesh generation challenges: A commercial software perspective, in: 23rd AIAA Computational Fluid Dynamics Conference, p. 3790.

T. Michal, Development of an anisotropic solution adaptive meshing tool for production aerospace applications, in: Sixth Workshop on Grid Generation for Numerical Computations.

C. Lock, O. Hassan, R. Sevilla, J. Jones, Meshing using neural networks for improving the efficiency of computer modelling, Engineering with Computers 39 (2023) 3791–3820.

P. L. George, H. Borouchaki, F. Alauzet, P. Laug, A. Loseille, D. Marcum, L. Maréchal, Mesh generation and mesh adaptivity: Theory and techniques, in:

E. Stein, R. de Borst, T. J. R. Hughes (Eds.), Encyclopedia of Computational Mechanics Second Edition, volume Part 1 Fundamentals, John Wiley & Sons, Ltd., Chichester, 2017.

A. Loseille, A. Dervieux, F. Alauzet, Fully anisotropic goal-oriented mesh adaptation for 3D steady Euler equations, Journal of Computational Physics 229 (2010) 2866–2897.

A. Chang-Hoi, L. Sang-Soo, L. Hyuek-Jae, L. Soo-Young, A self-organizing neural network approach for automatic mesh generation, IEEE transactions on magnetics 27 (1991) 4201–4204.

S. Alfonzetti, A neural network generator for tetrahedral meshes, IEEE transactions on magnetics 39 (2003) 1650–1653.

Z. Zhang, P. K. Jimack, H. Wang, MeshingNet3D: Efficient generation of adapted tetrahedral meshes for computational mechanics, Advances in Engineering Software 157 (2021) 103021.

K. Huang, M. Krügener, A. Brown, F. Menhorn, H. J. Bungartz, D. Hartmann, Machine learning-based optimal mesh generation in computational fluid dynamics, arXiv preprint arXiv:2102.12923 (2021).

J. Yang, T. Dzanic, B. Petersen, J. Kudo, K. Mittal, V. Tomov, J.-S. Camier, T. Zhao, H. Zha, T. Kolev, et al., Reinforcement learning for adaptive mesh refinement, in: International Conference on Learning Representations.

J. G. Wallwork, J. Lu, M. Zhang, M. D. Piggott, E2N: error estimation networks for goal-oriented mesh adaptation, arXiv preprint arXiv:2207.11233 (2022).

K. Tlales, K. E. Otmani, G. Ntoukas, G. Rubio, E. Ferrer, Machine learning adaptation for laminar and turbulent flows: applications to high order discontinuous Galerkin solvers, arXiv preprint arXiv:2209.02401 (2022).

C. Lock, O. Hassan, R. Sevilla, J. Jones, Predicting the near-optimal mesh spacing for a simulation using machine learning, in: E. Ruiz-Gironés, R. Sevilla, D. Moxey (Eds.), SIAM International Meshing Roundtable 2023, Springer Nature Switzerland, 2024, pp. 115–136.

K. J. Fidkowski, G. Chen, Metric-based, goal-oriented mesh adaptation using machine learning, Journal of Computational Physics 426 (2021) 109957.

S. Sanchez-Gamero, O. Hassan, R. Sevilla, A machine learning approach to predict near-optimal meshes for turbulent compressible flow simulations, International Journal of Computational Fluid Dynamics 38 (2024) 221–245.

P. J. Frey, P.-L. George, Mesh generation: application to finite elements, Iste, 2007.

M. T. Hagan, H. B. Demuth, M. Beale, Neural network design, PWS Publishing Co., Stillwater, 1997.

D. P. Kingma, J. Ba, ADAM: A method for stochastic optimization, arXiv preprint arXiv:1412.6980 (2014).

J. Peraire, J. Peiro, K. Morgan, Adaptive remeshing for three-dimensional compressible flow computations, Journal of Computational Physics 103 (1992) 269–285.

O. C. Zienkiewicz, J. Z. Zhu, The superconvergent patch recovery and a posteriori error estimates. part 1: The recovery technique, International Journal for Numerical Methods in Engineering 33 (1992a) 1331–1364.

O. C. Zienkiewicz, J. Z. Zhu, The superconvergent patch recovery and a posteriori error estimates. part 2: Error estimates and adaptivity, International Journal for Numerical Methods in Engineering 33 (1992b) 1365–1382.

M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, et al., TensorFlow: a system for Large-Scale machine learning, in: 12th USENIX symposium on operating systems design and implementation (OSDI 16), pp. 265–283.

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 (2003) 101–114.

J. H. Halton, Algorithm 247: Radical-inverse quasi-random point sequence, Communications of the ACM 7 (1964) 701–702.

B. Vandewoestyne, R. Cools, Good permutations for deterministic scrambled Halton sequences in terms of L2-discrepancy, Journal of Computational and Applied Mathematics 189 (2006) 341–361.