ECOLE POLYTECHNIQUE

MSc&T STEEM

Academic Year 2023-24

LIPPI Matteo

INTERNSHIP REPORT

‹‹Surrogate modeling of external aerodynamic simulations using machine learning and POD ››

NON CONFIDENTIAL REPORT

Academic Internship Supervisor: Fawaz Massouh

Site Supervisor: Jacques Peter

Internship dates: 01/04/2024 – 31/08/2024

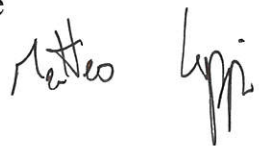
Declaration of Academic Integrity

Hereby I, Matteo Lippi, confirm that:

1. The results presented in this report are my own work.
2. I am the author of this report
3. I have not used the work of others without clearly acknowledging it, and quotations and paraphrases from any source are clearly indicated.

Matteo Lippi, 30/08/2024,

Signature



Abstract

This thesis investigates the development and application of reduced-order models (ROMs) for reconstructing parietal flows around civil aircraft configurations, conducted at the Office National d'Études et de Recherches Aéronautiques (ONERA). The research is divided into two main parts.

The first part utilises the XRF1 configuration to evaluate various ROM techniques, including k-NN interpolation, Proper Orthogonal Decomposition (POD), Isomap, Laplacian Eigenmaps, Diffusion Maps, Autoencoders, Variational Autoencoders, and both modewise and pointwise Multi-Layer Perceptrons (MLPs). Extensive testing and performance comparisons reveal MLP pointwise as the top performer, while Isomap shows notable advantages in dimensionality reduction and regression tasks, with performance comparable to well-trained MLP modewise models. The analysis of the results highlights the significance of hyperparameter tuning and suggests that Optimal Transportation methods and approaches incorporating mesh related information, such as Graph Neural Networks (GNNs), could further enhance predictive accuracy.

The second part of the thesis involves the creation of a comprehensive computational fluid dynamics (CFD) dataset aimed at supporting machine learning applications in aerodynamics. Several visualizations were performed to ensure and validate the high quality of the dataset.

Overall, the findings contribute to advancements in ROMs and machine learning for aerodynamic simulations, paving the way for future innovations in the field.

Context & Acknowledgements

The following work was conducted entirely at the Office National d'Études et de Recherches Aérospatiales (ONERA) in Châtillon. ONERA's mission is to advance aerospace research and ensure the effective dissemination of results both nationally and internationally, in collaboration with scientific and technical organizations. The research was carried out within the Aerodynamics, Aeroelasticity, and Acoustics Department (DAAA), specifically in the Demonstrations, Efficiency, Reliability, and Interoperability of Software (DEFI) research unit. DEFI focuses on developing numerical methods and technologies for high-performance simulations and conducts advanced demonstration calculations at the cutting edge of computational resources. This study benefited from the supervision of Jacques Peter and the support of Sébastien Heib, as well as the entire DEFI unit, led by Stéphanie Péron.

Contents

1 INTRODUCTION

Computational Fluid Dynamics (CFD) plays an essential role in the process of aerodynamic analysis and design. At the same time, CFD presents high computational costs, especially when dealing with a large space of flow conditions or a fine grid discretisation. Reduced Order Modelling, meaning the construction of a surrogate model of reduced computational complexity, offers a useful paradigm to tackle the above mentioned problem. Moreover, Reduced Order Models (ROMs) developed using Machine Learning techniques trained with a limited number of CFD simulation outputs have recently gained popularity [1].
This specific study focuses on the topic of ROMs of industrial CFD simulation outputs as function of the flow conditions, a topic of interest in the field of parametrised fluid dynamics. The structure of this document is as follows:

- Part 1 (§2 to 4) focuses on the behavior and performance of a series of ROMs for the reconstruction of parietal flows about the wing of a XRF1 civil aircraft configuration [2]. The study expands on the work done at Onera for the EG67/AG60 GARTEUR initiative (2019-2023) [1], introducing several new methods (including k_{NN} interpolation, POD, Isomap, Laplacian Eigenmaps, Diffusion Maps, Autoencoder, Variational Autoencoder), implementing them in code, and performing extensive testing and performance comparison to gather new insights. Additionally, the study also takes into consideration already existing methods (MLP modewise and MLP pointwise) and carries out both implementation/performance improvements and further analysis on their behaviour. Finally, perspectives on future research are given in §6. Specific sections of Part 1 are organised as follows: §2 presents the selected database, §3 introduces the ROMs, and §4 displays the results and their relative analysis.

- Part 2 (§5) describes the contributions to the creation of a CFD dataset to be utilised for the development and performance evaluation of several Machine Learning activities, including the continuations of the work performed in Part 1.

- Finally, §6 presents conclusions and future perspectives.

2 XRF1 DATASET

The dataset used for comparing the regression methods was provided by Airbus-M as part of the AG60 Garteur group framework [1]. Various wing-body CFD simulations were performed by Airbus-M with the DLR TAU Code solver for a XRF1 model in a wind tunnel (the XRF1 configuration [2] is close to a long range wide body Airbus aircraft). Regardless of the specific geometric configuration (with or without vertical tail plane, horizontal tail plane, or sting), the final dataset consists of

$$C_p = \frac{p - p_\infty}{1/2 \rho_\infty v_\infty^2}$$

coefficients only on the wing of the aircraft. The three sets of simulation outputs provided were divided by INTA into four pairs of train-test data, in order to define as many regression tasks [1].

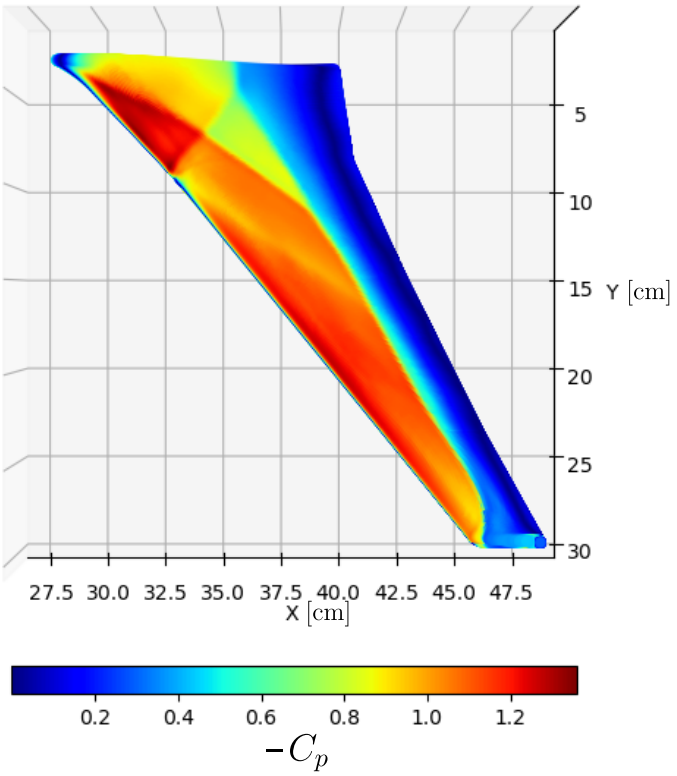


Figure 1: Wing $-C_p$ distribution (suction side) for the test case $M_\infty = 0.82$ and $AoA = 4.0°$.

For the current study, a dataset with consistent Reynolds number Re, varying far-field Mach number and angle of attack (M_∞, AoA), and identical location of data points on the wing for all flow conditions was selected. Specifically, $N = 112926$ C_p values are available for $n_{tot} = 426$ flow conditions. These conditions were divided into n_t=408 for training the regression models, and n_v=18 for testing their accuracy. Figure 1 shows one of the selected test conditions, while Figure 2 shows the entire dataset Design of Experiments(DOE) and its train-test split.

For the sake of generality, in the following, a flow condition defined by the pair (M_∞, AoA) is denoted by $\mathbf{p}$ (parameters of the regression). A C_p value is denoted by y, a C_p distribution is denoted by $\mathbf{y}$ (a vector of size N) and the complete set of training C_p distributions is denoted by $\mathbf{Y}$ (a matrix of size $N \times n_t$). The coordinates of individual wall points, which appear in pointwise regressors, are denoted by $\mathbf{c}$.

As the data generating process of $\mathbf{y}$ is governed solely by the $\mathbf{p}$ 2-dimensional variable, it follows that all the N-dimensional data points $\mathbf{y}_i$ must lie on a 2-dimensional manifold $\mathcal{Y}$ embedded in $\mathbb{R}^N$. A visual representation of this concept is shown in Figure 3.

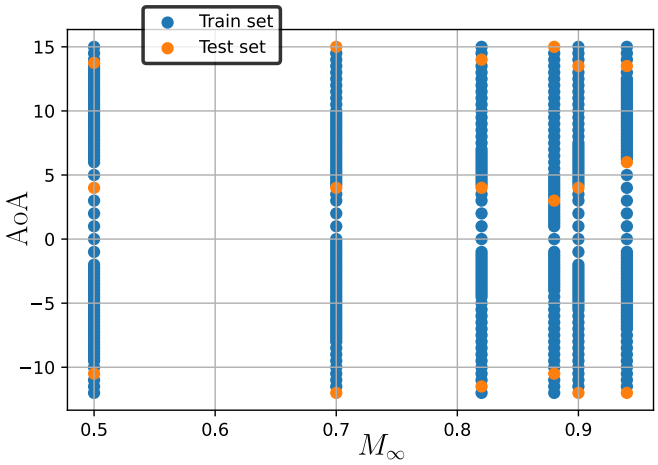


Figure 2: XRF1 flow conditions. Training and testing sets

3 REGRESSION METHODS

3.1 Reference methods

A first approach in predicting a local y^* or a complete wall distribution $\mathbf{y}^*$ for an unseen flow condition $\mathbf{p}^*$ may have been to use a classical interpolation technique such as Radial Basis Functions (RBF) or Kriging/Gaussian Process Regression (GP) [3]. Unfortunately, $N \times n_t$ is several orders of magnitude too high to search for pointwise regressors defining y as function of ($\mathbf{p}$, $\mathbf{c}$), and the linear system to solve would be of intractable size. Defining N classical (pointwise) surrogates for n_t data would very feasible although expensive, but the surrogates for neighboring points may not be consistent with each other. For these reasons, we consider two alternative methods as reference for the performance of more sophisticated ones.

3.1.1 Reference method 1: k_{NN} interpolation

The first method consists in computing $\mathbf{y}^*$ for unseen flow conditions $\mathbf{p}^*$, by performing a linear combination of the $\mathbf{y}_i$ vectors corresponding to the $k_{\mathcal{P}}$ nearest neighbours (k_{NN}) of $\mathbf{p}^*$ in the parameter space,

$$\mathbf{y}^* = f_{k_{NN}}(\mathbf{p}^*) = \frac{\sum_{i \in \mathcal{N}(\mathbf{p}^*)} w_i \mathbf{y}_i}{\sum_{i \in \mathcal{N}(\mathbf{p}^*)} w_i}. \tag{1}$$

Equation (1) formalises the methodology, denoting $\mathcal{N}(\mathbf{p}^*)$ the indexes of the k nearest neighbours of $\mathbf{p}^*$, and w_i the inverse of $||\mathbf{p}_i - \mathbf{p}^*||$, the Euclidean distance between $\mathbf{p}^*$ and one of the nearest points, $\mathbf{p}_i$, of the training set. This method does not involve any analysis of $\mathbf{y}$ and is hence unable to account for any non linearity of $\mathbf{y}$ as function of $\mathbf{p}$. It is only accurate (and even exact) if $\mathbf{y}$ is an affine function of $\mathbf{p}$, which is obviously not the case when the discrete (RANS) equations are involved to derive $\mathbf{y}$ from $\mathbf{p}$.

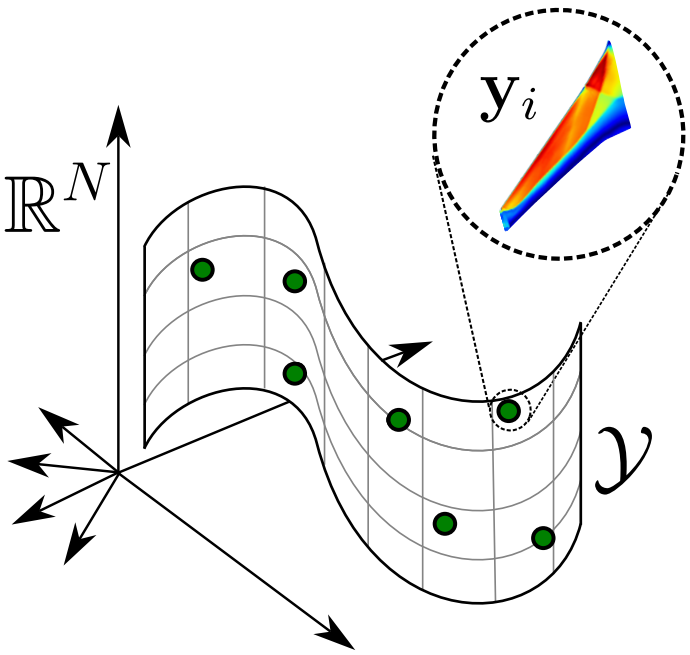


Figure 3: N-dimensional data points $\mathbf{y}_i$ lie on a 2-dimensional manifold $\mathcal{Y}$ embedded in $\mathbb{R}^N$.

3.1.2 Reference method 2: MLP modewise

A second straight-forward approach to the regression problem is provided by a technique from the field of Deep Learning called Multi Layer Perceptron (MLP). A MLP is a model that aims to approximate a function f. It achieves so by defining a parametrized regressor

$$\mathbf{y} = \hat{f}_{MLP}(\mathbf{p}, \boldsymbol{\theta})$$

and learning the values of the inner parameters $\boldsymbol{\theta}$ that result in the best approximation of f for the known data points $\mathcal{D} = (\mathbf{p}_i, \mathbf{y}_i)$ [4]. More precisely, the building block of an MLP is a "layer", defined by composing affine transformations $\mathbf{h}^{(l)} = \mathbf{W}\mathbf{h}^{(l-1)} + \mathbf{b}$ of the input (or previous layer data) with simple nonlinear functions σ such as *tanh* or $ReLU$ [4]. Repeating this building block multiple times (hence the name "Deep Learning") yields the MLP architecture, where the aforementioned inner parameters $\boldsymbol{\theta}$ are the weights, $\mathbf{W}$, and the shifts, $\mathbf{b}$, of the successive layers.
In order to learn suitable values of these inner parameters, an optimisation algorithm based on the gradient of an objective function is applied. A typical approach for the objective function definition in the machine learning literature is the method of maximum log likelihood [4]. This method consists in defining a probability distribution $p_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{p})$ for the model's predictions, whose mean is parametrised by the MLP output. The objective function is then set to the log probability that the known data points $\mathcal{D}$ were generated by the model $p_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{p})$. The optimal parameters $\boldsymbol{\theta}$ are considered to be the ones that maximise such objective function, which for $p_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{p}) \sim \mathcal{N}(f_{MLP}(\mathbf{p}), \sigma)$ is equivalent to minimisation of the Mean Square Error(MSE).
Taking advantage of the ease of differentiation of MLP through the use of the backpropagation algorithm [5], gradient-based optimisation tools are utilised in order to minimise MSE. It should be noted that gradient calculations of the objective function evaluated

on the entire $\mathcal{D}$ could become unfeasibly expensive. Because of this, estimators of the objective function value and gradients are constructed by evaluating them on random subsets of $\mathcal{D}$ denoted by the name of *batch*.
A visualisation of the MLP architecture, in the case of a network with one hidden layer, is presented in Figure 4.

For this specific study, a MLP was trained to map the $\mathcal{Y}$ pressure distribution space

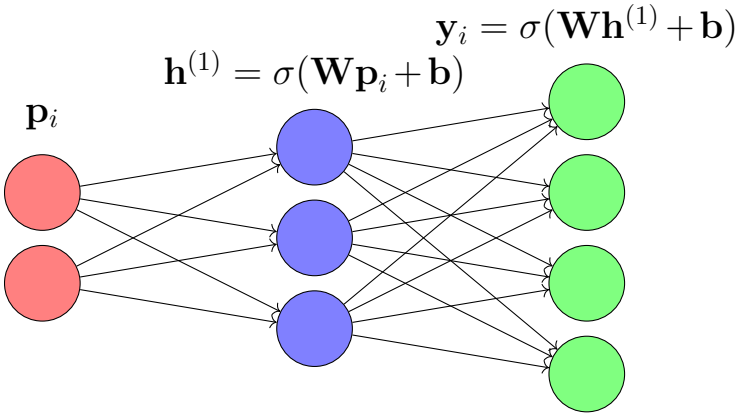


Figure 4: MLP architecture with a single hidden layer.

from the $\mathcal{P}$ parameter space. This involves a very large number of weights between the last hidden layer and the $\mathbf{y}$ output layer, but is sustainable with current CPUs. A visual representation of the model defined above is presented in Figure 5. (See §3.5 for a pointwise MLP.)

3.2 Dimensionality reduction coupled with regression

A different approach to the regression problem consists in initially searching for a low dimensional representation $\mathbf{z}_i \in \mathbb{R}^r$ ($r << N$), obtained by compressing the high dimensional pressure distributions $\mathbf{y}_i \in \mathbb{R}^N$ using a dimensionality reduction function $g(\mathbf{y}_i)$. The advantage of this pre-processing step is that in $\mathcal{Z} \subset \mathbb{R}^r$, also known as the latent space, performing interpolation $\mathbf{z}^* = f(\mathbf{p}^*)$ with regular methods (RBF, GP...) becomes feasible. After interpolation, the data is mapped back to the high dimension using a reverse mapping function $\mathbf{y}^* = h(\mathbf{z}^*)$. The entire modelling process is summarized in Figure 6.

For several methods, it is necessary to perform some type of interpolation in the high dimensional space $\mathcal{Y}$ from the latent space $\mathcal{Z}$, as no exact reverse mapping h^* is defined by the mapping process. The needed additional interpolation $h_{int}(\mathbf{z}^*) : \mathbb{R}^r \to \mathbb{R}^N$ may make the initial data compression g seem redundant, but performing dimensionality reduction can still be beneficial: specific structural features of the high dimensional data $\mathbf{y}$ are preserved in the low dimensional representation $\mathbf{z}$ and can be exploited to design more precise interpolation algorithms. This idea is applied in §3.4.1 and §3.4.2.

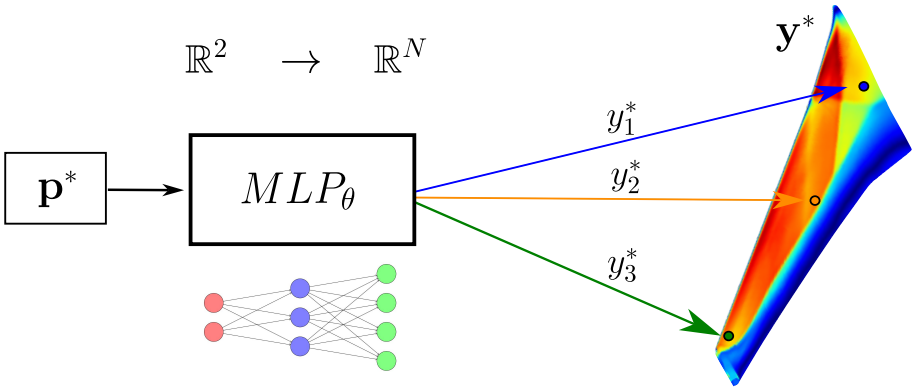


Figure 5: MLP Modewise model. Flow conditions $\mathbf{p}^*$ are mapped to their corresponding pressure distributions $\mathbf{y}^*$.

3.3 Linear dimensionality reduction (Proper Orthogonal Decomposition) plus Interpolation

Proper Orthogonal Decomposition (POD) [6] is a standard dimensionality reduction technique in fluid mechanics. To apply POD, the output matrix $\mathbf{Y} \in \mathbb{R}^{N \times n_t}$ is first processed by subtracting from each column (which represents a C_p distribution) the average C_p distribution $\mathbf{y}_b = {n_t}^{-1}\mathbf{Y}\mathbf{1}_{n_t}$ over all flow conditions (where $\mathbf{1}_{n_t}$ is a n_t dimensional column vector of ones). This preprocessing step yields the matrix $\bar{\mathbf{Y}} = \mathbf{Y}\mathbf{H}$, where $\mathbf{H} = \mathbf{I}_{n_t} - {n_t}^{-1}\mathbf{J}_{n_t}$ is a centering matrix and $\mathbf{J}_{n_t}$ is a $n_t \times n_t$ matrix of ones.
POD first consists in performing a Singular Value Decomposition (SVD) of the preprocessed matrix $\bar{\mathbf{Y}}$:

$$\bar{\mathbf{Y}} = \mathbf{U}\ \mathbf{\Sigma}\ \mathbf{Z}^T \tag{2}$$

where Σ is a (N, n_t) matrix with only diagonal and positive non-zero entries, and $\mathbf{U}, \mathbf{Z}^T$ are two orthonormal matrices of respective sizes (N, N) and (n_t, n_t). Subsequently, downsizing each matrix to only contain the components linked with the biggest r singular values yields a reduced rank-r representation of $\bar{\mathbf{Y}}$.

$$\bar{\mathbf{Y}}_r = \sum_{k=1}^{r} \sigma_k \mathbf{u}_k \mathbf{z}_k^t = \mathbf{U}_r \mathbf{\Sigma}_r \mathbf{Z}_r^T \tag{3}$$

where σ_k represents a diagonal element of Σ_r (a (r, r) matrix), $\mathbf{u}_k$ represents a column vector of $\mathbf{U}_r$ (a (N, r) matrix), and $\mathbf{z}_k^t$ represents a row vector of $\mathbf{Z}_r^T$ (a (r, N) matrix). POD can be interpreted in two fundamental ways:

1. *Physics:* POD aims to extract the set of most significant pressure distribution 'modes' $\mathbf{u}_k$ from the available centered data $\bar{\mathbf{Y}}$; such 'modes' can then be used as a basis to represent the original data.

2. *Manifold Learning:* POD performs a projection of the original manifold $\mathcal{Y}$ onto a r-dimensional plane that is optimal in the Frobenius norm sense, meaning that the reconstruction error $\|\bar{\mathbf{Y}} - \bar{\mathbf{Y}}_r\|_F$ is minimised. The columns $\mathbf{u}_k$ of $\mathbf{U_r}$ represent the basis in the r-dimensional plane, while the i-th column of $\mathbf{Z_r}^T$ corresponds to the latent representation (the coordinates) of the distribution $\mathbf{y}_i - \mathbf{y}_b$ on the r-dimensional plane.

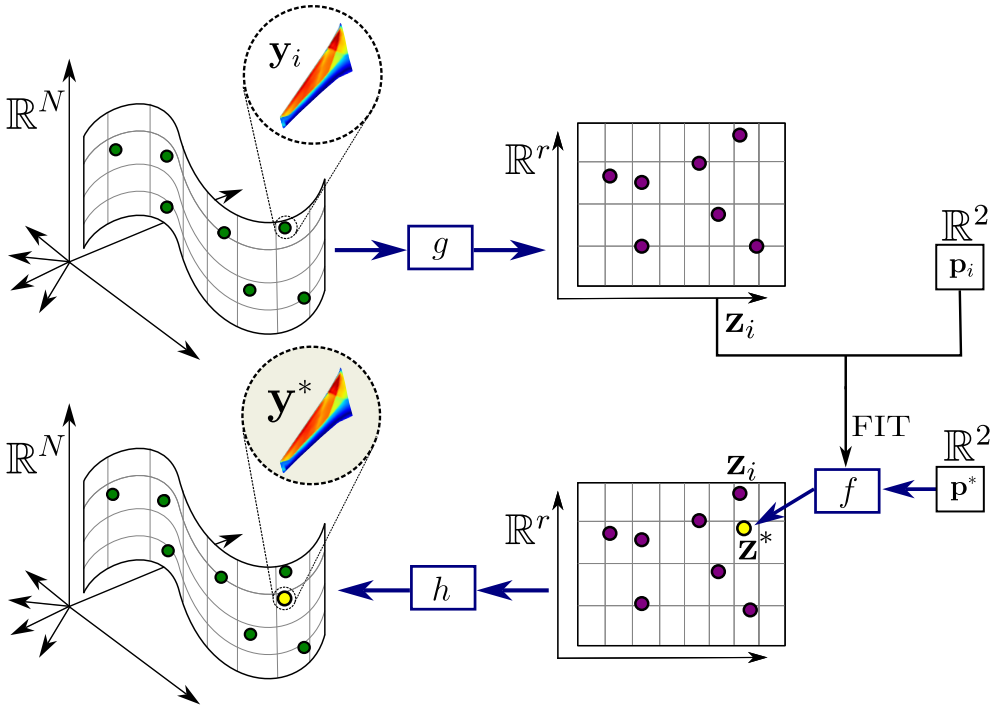


Figure 6: General principle of ROM involving dimensionality reduction.

Following the blueprint of Figure 6, once the latent representation $\{\mathbf{z}_i\}$ is extracted, r interpolators (GP, RBF or any other surrogate) are used to define all r latent coordinates $\mathbf{z}^*$ of an unseen parameter $\mathbf{p}^*$. Finally, the high dimensional $\mathbf{y}^*$ is reconstructed from its latent representation by substituting $\mathbf{z}^*$ in Equation (3).

$$\mathbf{y}^* = \mathbf{y}_b + \mathbf{U}_r \mathbf{\Sigma}_r \mathbf{z}^* \tag{4}$$

POD is a powerful technique that yields an exact (lossless) embedding when the $\{\mathbf{y}_i\}$ lie in a r-dimensional affine space. At the same time, its weakness lies in the linear nature of the projection it applies. For applications in aerodynamics, this does not usually pose a problem in the subsonic regime. On the other hand, when strong nonlinearities appear in the transonic regime, $\mathcal{Y}$ becomes unfit to be represented by a linear projection.

3.4 Non-Linear dimensionality reduction coupled with regression

3.4.1 Isomap

A better fitting method for more complex manifold geometries is Isomap [7], which can be intuitively understood as an attempt to isometrically "unwrap" twisted manifolds, like the ones appearing in Figure 3 and 7. From a dimensionality reduction point of view, the key idea behind Isomap is that, if the data lies on a $\mathcal{Y}$ which is not an affine space, the Euclidean distance does not represent a reliable measure of similarity (see dotted line of Figure 7). Isomap solves this problem by making use of the geodesic distance, which guarantees a more accurate measure of similarity (solid line of Figure 7).

More specifically, the Isomap algorithm consists of three steps:

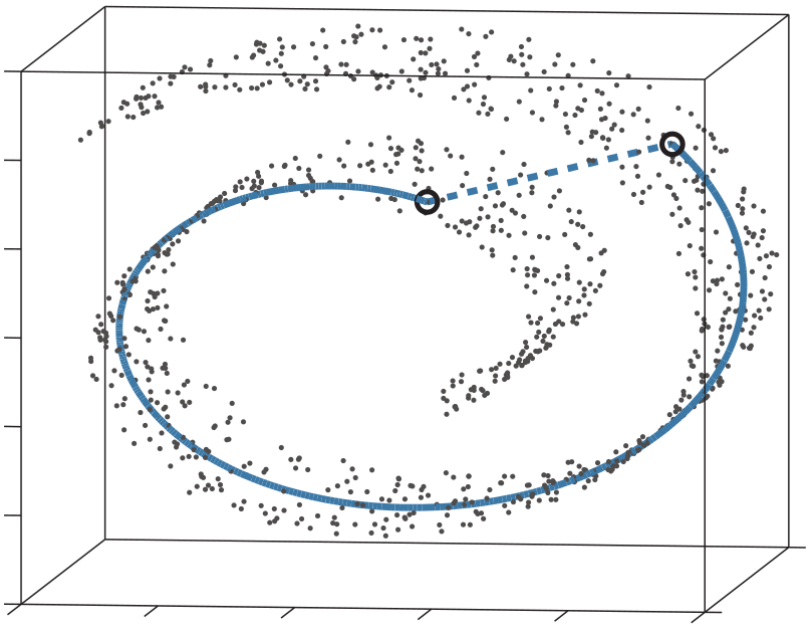


Figure 7: Euclidean and geodesic distances for two points sampled from a $\mathcal{Y}$ which is not an affine space. Adapted from [7].

1. Constructing the k_{NN} graph of the data points $\{\mathbf{y}_i\}$.
2. Constructing the matrix $\mathbf{D}_G$ by approximating geodesic distances $D_{G_{ij}} = d_G(\mathbf{y}_i, \mathbf{y}_j)$ between data points on $\mathcal{Y}$ as the shortest path in their k-NN graph, using methods such as the Dijkstra algorithm.
3. Applying classical Multi Dimensional Scaling (MDS) [8] to the matrix $\mathbf{D}_G$, in order to obtain a low dimensional embedding $\{\mathbf{z}_i\}$ in $\mathbb{R}^r$.

In MDS the matrix of scalar products $\mathbf{B}$ is first computed from $\mathbf{D}_G$ by:

$$\mathbf{B} = -\frac{1}{2}\mathbf{H}(\mathbf{D}_G \odot \mathbf{D}_G)\mathbf{H} \tag{5}$$

where $\odot$ is the Hadamard product. Afterwards, similarly to POD, diagonalising $\mathbf{B} = \mathbf{Z}\mathbf{\Lambda}\mathbf{Z}^T$ and downsizing each matrix to only include the r most significant components yields a low dimensional embedding, given by the columns $\mathbf{z}_i$ of $\mathbf{\Lambda}_r^{1/2}\mathbf{Z}_r^T$.
MDS guarantees that the matrix $\mathbf{D}_Z$, whose entries are given by $D_{Z_{ij}} = ||\mathbf{z}_i - \mathbf{z}_j||$, is the best rank r approximation of $\mathbf{D}_G$, minimizing $\|\mathbf{D}_G - \mathbf{D}_z\|_F$. This justifies the name "Isomap": the algorithm learns a discrete low dimensional embedding $\{\mathbf{z}_i\}$ of $\mathcal{Y}$ that has the property of being as isometric as possible with respect to the original manifold.
After having obtained the latent coordinates $\{\mathbf{z}_i\}$ of the training data, the interpolation procedure is carried out in a standard way (using GP, RBF...) to obtain $\mathbf{z}^*$. When having to reverse the mapping to obtain $\mathbf{y}^*$, the encoding of the data into $\mathbf{D}_G$ makes it impossible to derive an explicit analytical form such as the one of POD in (4). Two alternative solutions for h are explored:

1. A first approach, based on [9], takes profit of the approximate isometry between the latent space $\mathcal{Z}$ and the output space $\mathcal{Y}$. It consists in reporting in the output space a k-NN interpolation or a first-order Taylor formula satisfied in the latent space. In the former case, an optimal linear combination is searched for:

$$Min_{\ w^*} \quad \|\mathbf{z}^* - \sum_{i \in \mathcal{N}(\mathbf{z}^*)} w_i^* \mathbf{z}_i\| + \lambda \sum_{i \in \mathcal{N}(\mathbf{z}^*)} w_i^{*2} \tag{6}$$

$$s.t. \sum_{i \in \mathcal{N}(\mathbf{z}^*)} w_i^* = 1 \tag{7}$$

where the regularisation term in (6) is required as multiple solutions exist when the number of elements in $\mathcal{N}(\mathbf{z}^*)$ is greater than r, and $\mathcal{N}(\mathbf{z}^*)$ represents the indexes of the $k_{\mathcal{Z}}$ nearest neighbours of $\mathbf{z}^*$. After solving (6) (7), $\mathbf{y}^*$ is predicted using the w_i^* weights,

$$\mathbf{y}^* = \sum_{i \in \mathcal{N}(\mathbf{z}^*)} w_i^* \mathbf{y}_i, \tag{8}$$

consistently with local isometry.

2. A deep learning based method may be used to map the $\mathbf{z}_i$ to the $\mathbf{y}_i$, discarding at this stage the isometry property.

It is important to note that, as per Gauss's Theorem Egregium, exact isometric embedding of $\mathcal{Y}$ in dimension $r = 2$ exists only for manifolds presenting zero Gaussian curvature at every point. The $\mathcal{Y}$ shown in Figures 6 and 7 represent two of these specific cases. For any other case that presents non-zero Gaussian curvature(for example a paraboloid), Isomap can only provide the Euclidean representation of $\mathcal{Y}$ that is closest to an isometry.

3.4.2 Laplacian Eigenmaps and Diffusion Maps

While Isomap focuses on achieving global isometry between $\mathcal{Y}$ and $\mathcal{Y}$, other methods aim to preserve local structure, following the hypothesis that high correlations (i.e. small distances) represent the only meaningful information on the data set. This is the case of Laplacian Eigenmaps [10], a technique that works by minimizing the quantity:

$$\sum_{i,j} \|\mathbf{z}_i - \mathbf{z}_j\|^2 W_{ij} \tag{9}$$

where $W_{ij} = e^{-\frac{\|\mathbf{y}_i - \mathbf{y}_j\|}{t}}$ is the heat kernel if $\mathbf{y}_i$ and $\mathbf{y}_j$ are connected in the symmetric k-NN graph $\mathcal{G}$, and $W_{ij} = 0$ otherwise. Equation (9) minimises the distance between two points in $\mathcal{Z}$ with increasing importance the closer the corresponding points are in $\mathcal{Y}$.
With reasonable constraints to remove scaling factors, minimizing (9) is equivalent to finding the matrix of eigenvectors $\mathbf{Z}_r$ corresponding to the r smallest eigenvalues (excluding the first one, which is always zero in the case of a connected graph) of the following generalised eigenvalue problem:

$$\mathbf{L}\mathbf{z} = \lambda \mathbf{D}\mathbf{z} \tag{10}$$

where $\mathbf{D}$ is the degree matrix of $\mathcal{G}$, $\mathbf{W}$ is the matrix of W_{ij} and $\mathbf{L} = \mathbf{D} - \mathbf{W}$ is the graph Laplacian. The low dimensional embedding $\{\mathbf{z}_i\}$ satisfying (9) is given by the rows of $\mathbf{Z}_r$.

The efficiency of Laplacian Eigenmaps comes from its connection with the Laplace-Beltrami operator $\mathcal{L}$, which acts on manifolds and of which $\mathbf{L}$ is a discretisation in the case of data points uniformly distributed on the manifold surface. In the (frequent) case of non uniformly distributed data points, $\mathbf{L}$ is no longer a good discretisation of $\mathcal{L}$, and a generalisation of Laplacian Eigenmaps called Diffusion Maps [11] can be utilised to recover a correct discretisation. Diffusion maps defines a diffusion matrix as:

$$\mathbf{L}^{(\alpha)} = \mathbf{D}^{-\alpha}\mathbf{L}\mathbf{D}^{-\alpha} \tag{11}$$

and a new degree matrix $\mathbf{D}^{(\alpha)}$ such that $\mathbf{D}^{(\alpha)}_{ii} = \sum_j L^{(\alpha)}_{ij}$. The parameter α controls the influence of the data points density, with $\alpha = 0$ simplifying to $\mathbf{L}$, and $\alpha = 1$ defininig the proper discretisation of $\mathcal{L}$. Substituting $\mathbf{L}^{(\alpha)}$ and $\mathbf{D}^{(\alpha)}$ in (10) yields the Diffusion maps embedding.
In both cases, after a latent representation is obtained, the blueprint of Figure 6 is applied. Following interpolation, the reverse mapping procedures presented in §3.4.1 for Isomap are utilised.

3.4.3 Autoencoder and Variational Autoencoder

An autoenconder (AE) [12] is a specific type of MLP that can learn nonlinear low dimensional representations of a given dataset. An AE, denoted f_{AE}, can be thought of as the sequential application of two network sub-parts, divided by a low dimensional layer $\mathbf{z}$: an encoder network $\mathbf{z} = g_{AE}(\mathbf{y})$, and a decoder network $h_{AE}(\mathbf{z})$, so that $f_{AE}(\mathbf{y}) = h_{AE}(g_{AE}(\mathbf{y})) = h_{AE}(\mathbf{z})$. The working principle of an AE consists in training the network to reconstruct its own inputs:

$$\hat{\mathbf{y}}_i = f_{AE}(\mathbf{y}_i) = h_{AE} \circ g_{AE}(\mathbf{y}_i) \simeq \mathbf{y}_i$$

While a regular MLP would just learn the identity function, the presence of the $\mathbf{z}$ layer forces the AE to learn how to extract a low dimensional representation of $\mathbf{y}$, as the sequentiality of the layers means that the reconstruction process has to be based solely on $\mathbf{z}$. A visual representation of the AE architecture is presented in Figure 8.
The high complexity of the loss landscape of an AE makes the training process prone to

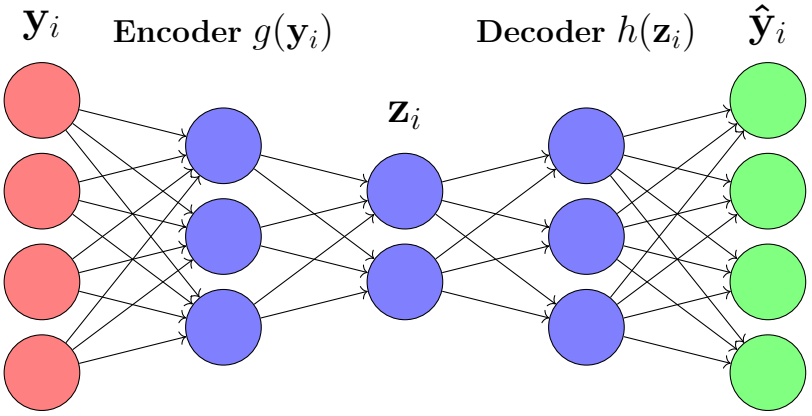


Figure 8: Autoencoder architecture.

learning suboptimal parameters. Additionally, as many low dimensional representations

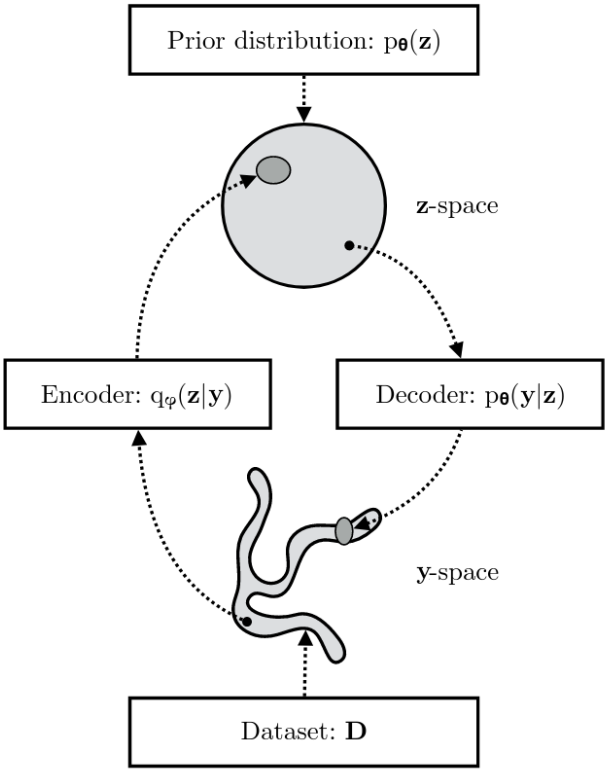


Figure 9: VAE working principle. Adapted from [14]

may exist, an AE could learn a specific $\mathbf{z} = g_{AE}(\mathbf{y})$ that is not optimal for interpolation in the latent space.

A Variational Autoencoder (VAE) [13] is a probabilistic generalisation of the AE that, amongst (many) other features, achieves a regularisation of the latent space. A VAE works by considering the joint probability distribution $p_{\boldsymbol{\theta}}(\mathbf{y}, \mathbf{z}) = p_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{z})p(\mathbf{z})$, whose parameters $\boldsymbol{\theta}$ are given by a MLP (equivalent to the decoder network of the AE architecture). VAE attempts to solve the Bayesian inference problem of finding

$$p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{y}) = \frac{p_{\boldsymbol{\theta}}(\mathbf{y}, \mathbf{z})}{p_{\boldsymbol{\theta}}(\mathbf{y})},$$

which is usually intractable as the integral $p_{\boldsymbol{\theta}}(\mathbf{y}) = \int p_{\boldsymbol{\theta}}(\mathbf{y}, \mathbf{z}) d\mathbf{z}$ does not have an analytic solution or efficient estimator. The VAE solution is obtained by approximating $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{y})$ with a probability distribution $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{y})$, parametrised by a second MLP (equivalent to the encoder of the AE). A diagram showing the VAE working principles is presented in Figure 9.

The setup of Figure 9 allows a VAE to effectively learn an approximation $q_{\boldsymbol{\phi}}(\mathbf{z})$ of the probability distribution of the latent variable $\mathbf{z}$, thus generalising the latent space concept of an AE. The characteristics of $q_{\boldsymbol{\phi}}(\mathbf{z})$ are strongly influenced by the choice of the prior $p(\mathbf{z})$. More specifically, in the case of $p(\mathbf{z}) \sim \mathcal{N}(0, \mathbf{I})$, the learned probability distribution is regularised to be smooth, bound by a r-ball of center zero, and to have mutually independent $\mathbf{z}$ coordinates. These are all desirable features for the task of interpolating in the space of $\mathbf{z}$.

VAE is trained by minimising an objective function known as Evidence Lower Bound or ELBO, given by:

$$\mathcal{L}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\mathbf{y}) = \log p_{\boldsymbol{\theta}}(\mathbf{y}) - D_{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{y})||p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{y})), \tag{12}$$

where the second term on the right is the Kullback-Liebler divergence. Kullback-Liebler divergence is defined by

$$D_{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{y})||p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{y})) = \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{y})}[\log[\frac{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{y})}{p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{y})}]],$$

which quantifies the difference between two probability distributions.
As $D_{KL} \geq 0$, maximising the ELBO has a two-fold effect. On one hand, it is equivalent to approximately maximising the (intractable) log likelihood $\log p_{\boldsymbol{\theta}}(\mathbf{y})$, which trains the decoder part of the VAE. On the other hand, it is also equivalent to approximately minimising the second term in (12), which improves the accuracy of $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{y})$, training the encoder part of the VAE. Using specific assumptions on the forms of the different probability distributions (like the ones presented in §4.1.2), (12) can be further manipulated to yield a loss function composed by the sum of a MSE term and a $D_{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{y})||p(\mathbf{z}))$ term. This loss function is the one utilised in the implemented models.
In order to predict $\mathbf{y}^*$, the blueprint of Figure 6 is followed for both AE and VAE. Interpolation on the $\mathbf{z}_i$ is carried out in the standard way using RBF; subsequently the decoder network h_{AE} is applied to $\mathbf{z}^*$ to obtain $\mathbf{y}^*$ (given by $\mathbb{E}[\mathbf{y}|\mathbf{z} = \mathbf{z}^*]$ in the VAE case).

3.5 Pointwise approach

The entirety of the methods presented above make use of pressure distributions $\mathbf{y}_i$ "flattened" as N-dimensional vectors. This preprocessing procedure reduces the original data, in the sense that the spatial position of each pressure extraction point y_n in the physical world is lost; the only constraint in the modewise methods of §3.1 to §3.4 being to always order consistently the y values.
To aid the regression model to better generalize to unseen flow conditions, it can be made more descriptive by incorporating geometric information: the coordinates $\mathbf{c}$ of the pressure points may be joined to the parameters $\mathbf{p}$ as inputs of so-called "pointwise" methods – see also Deepsets architecture [15]. Locations and flow conditions are then seen as equivalent inputs from a Machine Learning point of view, whereas from a Mechanical point of view they definitively are not (part of the flow conditions can possibly be discarded from a study, whereas the complete wing geometric domain is always needed).
Practically, the method consists in training an MLP regression model $f_{MLP} : \mathbb{R}^5 \to \mathbb{R}$ that predicts each y_n from its coordinates $\mathbf{c}_n$ and the parameters $\mathbf{p}$. A visual representation of the MLP pointwise model is presented in Figure 10.

3.6 Hyperparameter tuning for Deep Learning architectures

In the case of POD, Isomap, Laplacian Eigenmaps and Diffusion maps, hyperparameters are straightforwardly found performing a grid search. On the other hand, for Deep Learning methods, hyperparameter tuning is a significantly more complex problem, the results of which can strongly affect model performance. Being an important component in the replicability of the results, the general hyperparameter tuning workflow is described here,

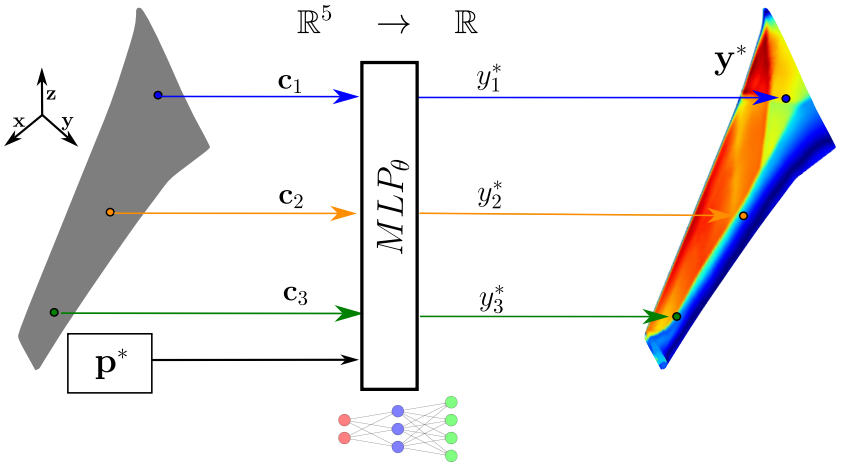


Figure 10: MLP Pointwise model. Flow conditions and point coordinates $(\mathbf{p}^*, \mathbf{c}_n^*)$ are mapped to the corresponding C_p value y_n^*.

while additional specifications and results are presented in §4.1.2.
As all Deep Learning methods considered are based on the MLP architecture, the types of hyperparameters to be optimised are shared amongst all methods. Hyperparameters could be divided in architectural hyperparameters (network depth D, number of neurons for each layer $s^{(l)}$, choice of σ) and training process hyperparameters (learning rate lr, regularisation coefficient α, number of training epochs N_{epochs}, batch size B, early stopping).
Following the guidelines provided in [16], hyperparameters were divided in three categories, in relation to their role in the tuning process:

- *Scientific hyperparameters:* parameters whose effect on the model performance it is of interest to measure. For this specific study, the category included D and the choice of σ.

- *Nuisance hyperparameters:* parameters that need to be optimised over in order to fairly compare different values of the scientific hyperparameters. For this specific study, the category included $s^{(l)}$, lr, α.

- *Fixed hyperparameters:* parameters whose values remains unchanged for the entirety of the tuning process. For this specific study, the category included N_{epochs} and the choice of early stopping. The value of B was also considered to be a fixed hyperparameter, as according to [17], there appears to be no evidence that batch size can affect model performance.

After performing the categorisation presented above, several hyperparameter tuning studies were carried out using the Optuna framework [18], in order to gain insights on the effects of the different scientific hyperparameters on model performance.

4 RESULTS

After coding and training all the presented methods for the training set ($n_t = 408$ XRF1 wing pressure distributions), the R^2 scores,

$$R^2 = 1 - \frac{\sum_{v=1,j=1}^{n_v,N} (y_{v,j} - \hat{y}_{v,j})^2}{\sum_{v=1,j=1}^{n_v,N} (y_{v,j} - \overline{y})^2}$$

were calculated for the testing set ($n_v = 18$ distributions). Results are shown in Table 1. The scikit-learn library [19] was utilised to carry out the implementation of all methods; the Pytorch library [20] was also leveraged for the implementation of all deep learning methods. All RBF/GP interpolation was performed by using the SMT library [21]. (As usually done, all methods were applied to normalised input data.)

4.1 Method specific parameters

In this study, the hyperparameters of the ROMs have been optimized to get the best performance on the test set, establishing their best possible accuracy. In future studies, all hyperparameters shall be optimized using a reserved part of the training data, as done in –[1] §3.

4.1.1 Traditional Machine Learning methods

- *(Control case) k_{NN} regression:* The number of neighbours was optimised by grid search and chosen to be $k_{\mathcal{P}} = 8$.
- *Proper Orthogonal Decomposition:* $r = 307$ modes were extracted to satisfy the classical 99% criterion on the sum of the $\mathbf{\Sigma}$ eigenvalues. RBF and GP for interpolation from $\mathcal{P}$ to $\mathcal{Z}$ lead to equivalent results.
- *Isomap:* The latent dimension r, the number of neighbours $k_{\mathcal{Y}}$ in the k_{NN} graph construction, and the size of the reverse mapping neighbourhood $k_{\mathcal{Z}}$ were considered as parameters, and optimised using grid search. The selected values were $r = 5$, $k_{\mathcal{Y}} = 14$, and $k_{\mathcal{Z}} = 10$.
- *Laplacian Eigenmap and Diffusion Map:* The latter performed better, leading to its selection. For Diffusion Maps, the same parameters as those of Isomap were considered, and the grid search yielded $r = 60$, $k_{\mathcal{Y}} = 200$, $k_{\mathcal{Z}} = 13$.

4.1.2 Deep Learning methods

As presented in §3.6, several hyperparameter tuning studies were performed to improve model performance.
The first hyperparameter tuning study was carried out for MLP modewise. Specifically, for each combination of the scientific hyperparameters $D = [1, 2, 3, 4, 5]$ and $\sigma = [ReLU, tanh]$,

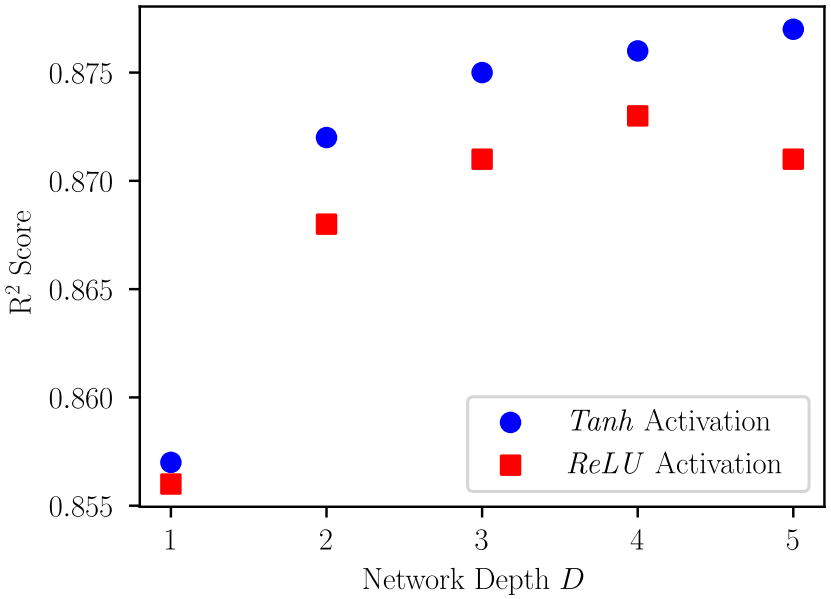


Figure 11: MLP modewise performance as a function of D and σ, when optimising over the remaining hyperparameters.

200 trials of random search over the nuisance hyperparameters were performed, in order to find their optimal combination. Afterwards, the performance of the optimal models for each (D, σ) was compared, and the results are shown in Figure 11. *ReLU* perfomed significantly worse, while also presenting a large training variance (the variation in perfomance observed between training runs that use the same model, but different random seeds), an order of magnitude higher compared to the one of the *tanh* model. High training variance is an undesirable feature as benchmarking model performance becomes significantly harder.

After completing this process and having identified the best (D, σ) combination for MLP modewise, a second study for AE and VAE was carried out. The study considered thesize of the bottleneck layer $\mathbf{z}$ as scientific hyperparameter, while using the same model architecture as MLP modewise for the decoder part, and a symmetric opposite for the encoder part. Once again, 200 trials of random search over the nuisance hyperparameters were performed, and the performance of optimal models as a function of the bottleneck size is shown in Figure 12.

All the selected hyperparameters that were used to obtain the results presented in Table 1 are listed below:

- *(Control case) MLP modewise:* 4 hidden layers of size $(384, 309, 249, 204)$ were used. It was observed that increasing the number of layers did result in a significant improvement in performance. The activation function was *tanh*. Adam optimiser was used to decrease a standard MSE loss with $L2$ weight regularisation (coefficient $\alpha = 10^{-4}/n_t$). The model was trained for 200 epochs, performing early stopping if the performance on the test set would not increase after 50 epochs.

- *Autoencoder and Variational Autoencoder:* The same architecture and hyperparam-

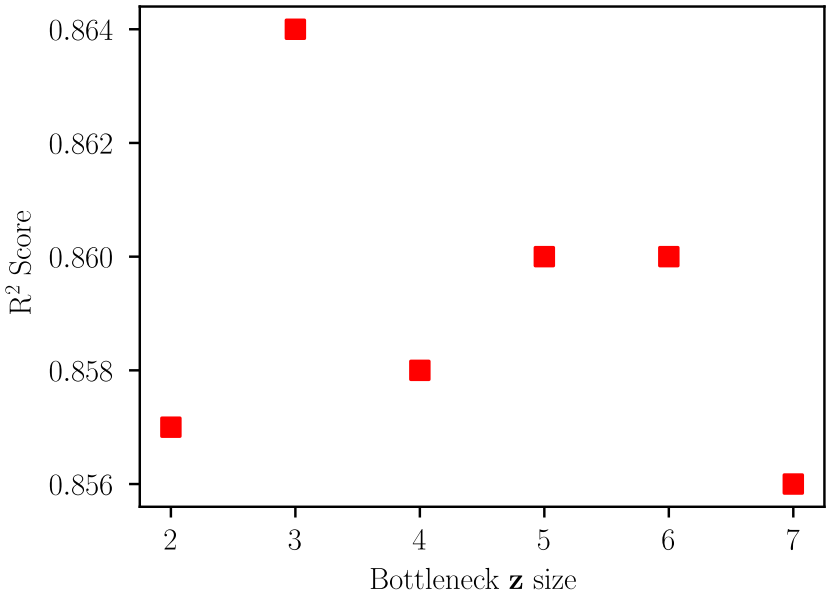


Figure 12: VAE performance as a function of $\mathbf{z}$, when fixing the architectural parameters to be the same as those best performing for MLP modewise and optimising over the remaining hyperparameters.

eters were used. In both cases, a 4-hidden-layer encoder with layer sizes $(384, 309, 249, 204)$ was selected, followed by one low dimensional layer of size r and a symmetrical decoder. The value of r was optimised using grid search leading to $r = 5$ for AE and $r = 3$ for VAE. Finally, the same loss function as in the MLP modewise case was utilised for the AE, while the classical Kullback–Leibler divergence term was added in the VAE case. Similarly to MLP modewise, both models were trained for 200 epochs, performing early stopping if the performance on the test set would not increase after 50 epochs. In the specific case of VAE, the $p_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{z})$ and $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{y})$ were chosen to be respectively $\mathcal{N}(h_{\boldsymbol{\theta}}(\mathbf{z}), \sigma^2\mathbf{I})$ and $\mathcal{N}(g_{\boldsymbol{\phi}1}(\mathbf{y}), g_{\boldsymbol{\phi}2}(\mathbf{y})\mathbf{I})$, where $g_{\boldsymbol{\phi}1}$ and $g_{\boldsymbol{\phi}2}$ are two encoders that only differ in the parametrisation of the last layer. Finally, the prior was taken to be $p(\mathbf{z}) \sim \mathcal{N}(0, \mathbf{I})$ as presented in §3.4.3.

- *MLP pointwise:* a 3-layer network with $(50, 55, 60)$ neurons was used, as no improvement was recorded with the use of more layers or neurons. The same optimisation and activation function as for MLP modewise were selected. As the training set for the method was very large (size $N \times n_t$), a division of the loss in batches of 100 was implemented. Training was performed with early stopping and it was noted that the model performance converges quite fast, reaching a $R^2 \simeq 0.90$ after a single epoch.

4.2 Performance Comparison

It is noteworthy that only Isomap, MLP modewise, and MLP pointwise perform above the very basic k_{NN} interpolation of outputs from the parameter space.
The underperformance of POD and the better performance of Isomap hints to the struc-

ture of $\mathcal{Y}$ being highly nonlinear. It is also noted that dimensionality reduction models with stricter priors on the latent space, such as Isomap, performed better than more general models such as AE, VAE or Diffusion Maps (which can, to an extent, be considered as a relaxation of the global isometry principle of Isomap). One of the reasons of such divide could also be the ease of training an Isomap model, which requires significantly fewer hyperparameters than AE or VAE .
Finally, the clear superiority of the MLP pointwise approach in terms of R^2 performance suggests that integrating various aspects of the physical process of interest in the regression model can be highly beneficial.

Table 1: R^2 values for each of the methods considered in the study

Method:	$\mathbf{R^2}$ **score**
(Control) k_{NN} Regression	0.865
(Control) MLP Modewise	0.876
POD	0.845
Isomap	0.876
Diffusion maps	0.857
AE	0.861
VAE	0.864
MLP Pointwise	0.923

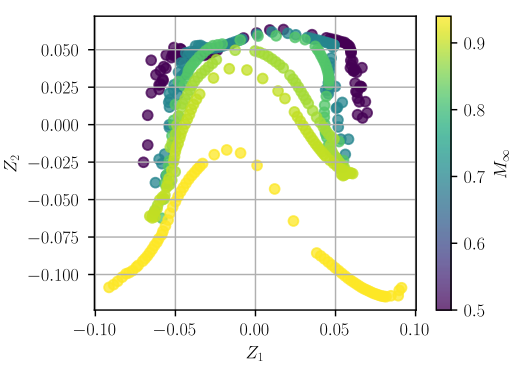
4.3 Visualisation of low dimensional representations for $r = 2$

While for many dimensionality reduction methods $r = 2$ is not an optimal value, visualising the $\{\mathbf{z}_i\}$ obtained by setting $r = 2$ can provide valuable insights on the structure of the latent spaces. Values of $\{\mathbf{z}_i\}$ generated by several methods from §3.4 are shown in Figure 13, with a colour scale indicating the respective M_∞ or AoA of each point.
It is important to highlight that, while naively a low dimensional representation such as the DOE of Figure 2 would appear desirable, the exact reconstruction of Figure 2 would not be an optimal result. The hope is to obtain a low dimensional embedding that resembles Figure 2, but is not isometric to Figure 2, as the additional value of dimensionality reduction techniques lies in bringing similarities that are present in $\mathcal{Y}$ into a low dimensional form $\mathcal{Y}$.
Generally, the generating parameter AoA is identified quite well by the majority of the dimensionality reduction methods, as it is shown in the case of Isomap in the subfigure *(c)*. On the other hand, a proper reconstruction of M_∞ appears to be more challenging. Both in the case of POD and Isomap, one dimensional structures of constant AoA can be observed. Interestingly, it can be noted that higher M_∞ corresponds to a more defined structure.
This effect could be linked to the appearance of characteristic shock structures at higher M_∞, making all the pressure distributions given by a specific M_∞ qtrongly correlated with each other. As the M_∞ decreases, the $\mathbf{z}_i$ embeddings linked with higher AoA be-

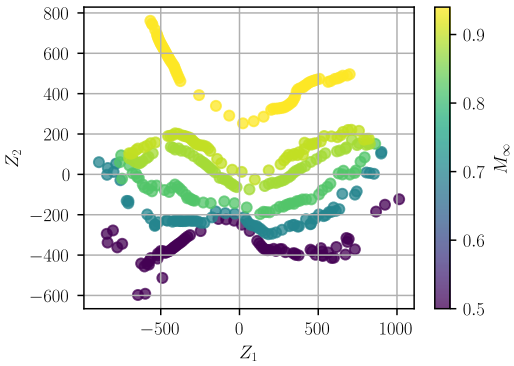
come increasingly mixed with each other, losing their one dimensional structure and thus becoming harder to interpolate. The main desirable feature of Isomap embeddings appears to be the separation of low M_∞ 1-dimensional structures, which become entangled together in the POD case.
Finally, comparing subfigure *(c)* with Figure 2 brings to light the added value of Isomap embedding. As an example, the case $M_\infty = 0.50$ and 0.70 can be taken into consideration. Figure 2 presents each pair of same-valued AoA points as equidistant. On the other hand, Isomap's representation clearly shows that the pairwise distance is an increasing function of $|AoA|$, reflecting the decrease in similarity of the corresponding $\mathbf{y}_i$.
Subfigures *(d)* to *(f)* show the embeddings obtained by AE and its variations. In general, the embeddings obtained by these Deep Learning techniques present less defined 1-dimensional structures. As it can be seen in subfigure *(d)*, the regular AE struggles to recover the original parametrisation of the dataset. VAE performs better, probably due to the influence of the Gaussian prior $p(\mathbf{z}) \sim \mathcal{N}(0, \mathbf{I})$ that favours the independence of the latent representation coordinates. Finally, subfigure *(f)* aims to show the effects that the latent space regularisation performed by the Gaussian prior has on the $\{\mathbf{z}_i\}$. It presents the case of a VAE where the influence of $p(\mathbf{z})$ has been artificially increased by $\beta = 30$ times (obtained by multiplying the Kullback–Leibler divergence term of the loss function by β): the embeddings are centered at $(0, 0)$, with points being more strongly concentrated towards the center.

4.4 Error visualisation for cruise condition $M_\infty = 0.82$, $AoA = 4.0°$

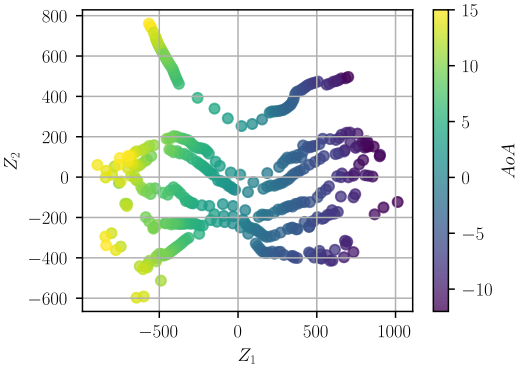
To further investigate the sources of error present in the generated predictions $\mathbf{y}^*$, a particular test case was selected for error visualisation. The error visualisation case, corresponding to the pressure distribution $\mathbf{y}$ generated by flow conditions $M_\infty = 0.82$ and $AoA = 4.0°$, was chosen for its virtue of being close to cruise conditions for the XRF1. Figure 14 shows the absolute value of the error $|\hat{\mathbf{y}} - \mathbf{y}|$ for predictions of the $M_\infty = 0.82$ and $AoA = 4.0°$ case which were generated by the various considered methods.
It can be seen that, for the best performing models, the error appears to be highly localised at the locations of highest gradient, where the shock waves are taking place. Even more interestingly, performance outside of the high gradient region does not degrade when using very basic models such as k_{NN} interpolation, which relies on a simple linear combination in $\mathcal{Y}$.



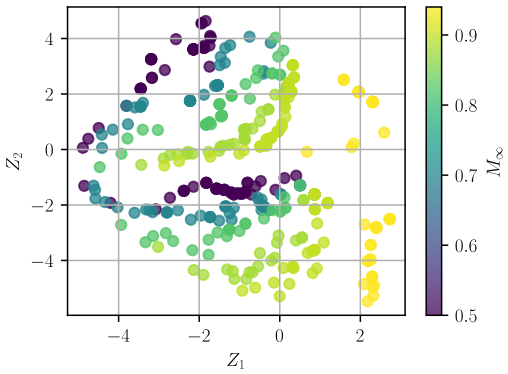
(a) POD



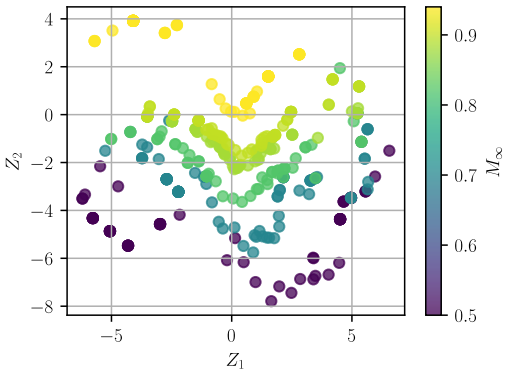
(b) Isomap with respect to M_∞



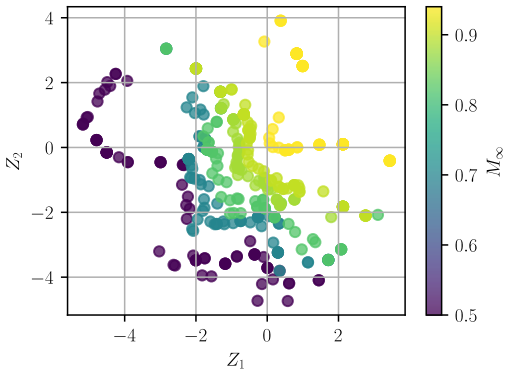
(c) Isomap with respect to AoA



(d) Autoencoder



(e) Variational Autoencoder



(f) Variational Autoencoder with increased influence of gaussian prior $p(\mathbf{z}) \sim \mathcal{N}(0, \mathbf{I})$

Figure 13: 2-dimensional embeddings generated by the dimensionality reduction methods from §3.4

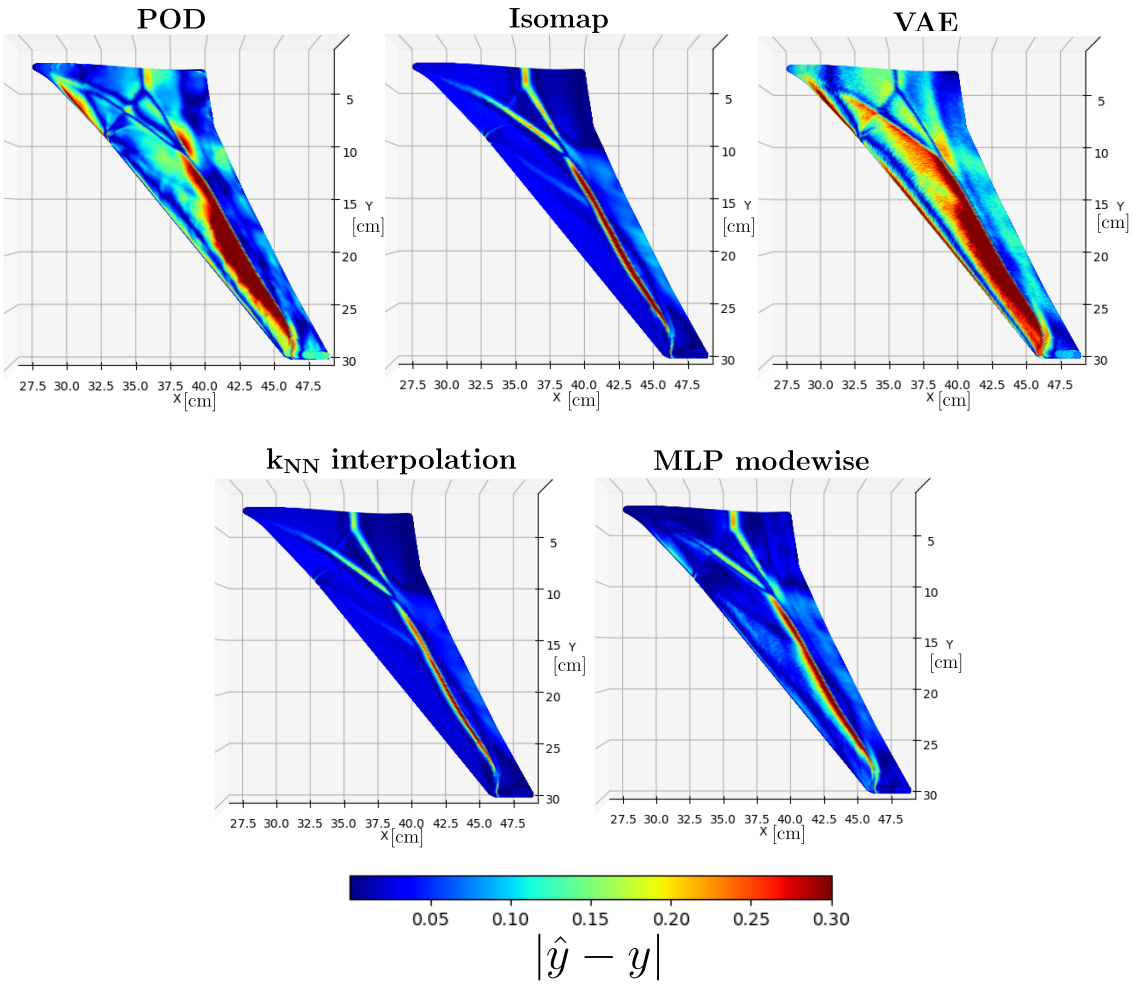


Figure 14: Absolute value of the error $|\hat{\mathbf{y}} - \mathbf{y}|$ for predictions of the $M_\infty = 0.82$ and $AoA = 4.0°$ case.

5 CONTRIBUTIONS TO THE CREATION OF A CRM-WBPN3 DATASET

5.1 Physical aspects

In order to enable several machine learning research projects, the creation of a suitable dataset has been carried out at ONERA in 2024. The dataset consists of parietal fields such as pressure coefficients C_p and friction coefficients C_f, about a civil aircraft with wing-body-pylon-nacelle (WBPN) configuration. These fields were obtained from CFD simulations with ONERA's large CFD code elsA [22].
The chosen configuration has been the NASA CRM (Common Research Model) [23], due to the following characteristics:

1. Public availability, to create no constraints on access or publication;

2. Intermediate complexity representative of a modern transport aircraft, with achievable controlled convergence at an acceptable cost;

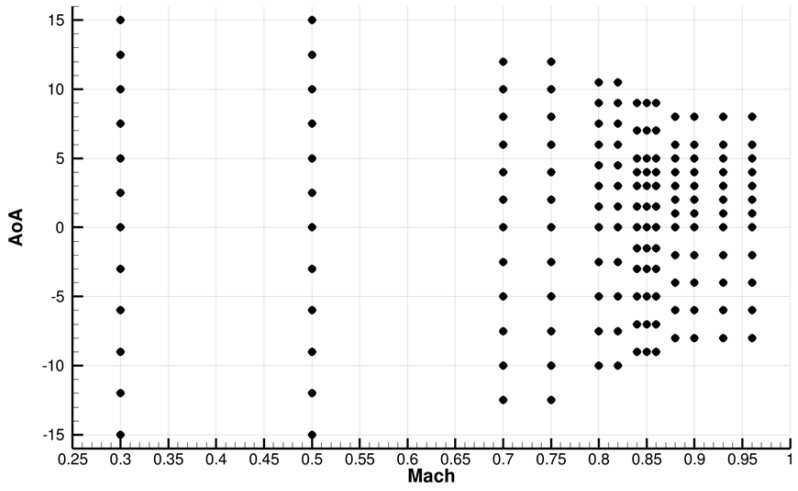


Figure 15: CRM-WBPN3 Design of Experiments.

3. Tested in wind tunnels by several aeronautical research organizations, to enable activities of comparison between calculations and experiments;

4. A CRM model is available at ONERA, and Wind Tunnel tests have already been performed.

The primary machine learning application of the dataset, stemming from the work conducted in the previous sections, is the prediction of parietal fields based on far-field conditions. Additionally, incorporating the available data on mesh normals and connectivity could further enhance predictive performance. Other envisioned machine learning activities also include integrating simulation data with experimental results and combining data from coarse and fine mesh simulations. The DOE was developed as to include the DOE of wind tunnel experiments, in order to enable comparisons between the two. More specifically, the results of each simulation are determined by four generating quantities: the M_∞, the AoA, the stagnation pressure p_i and the stagnation temperature T_i. The value of T_i is kept constant to a value of 310 K, while p_i can take three different values: 100000 Pa, 200000 Pa and 400000 Pa (atmospheric and two pressurised wind tunnels) , in order to achieve a range of Re(N.B. the simulations do not account for the wing aeroelasticity that would be impacted by these changes in total pressure). For each value of p_i, simulations for 13 different M_∞ (around the cruise M_∞) times 12 different AoA are performed. The specifications of the DOE are detailed in Figure 15.

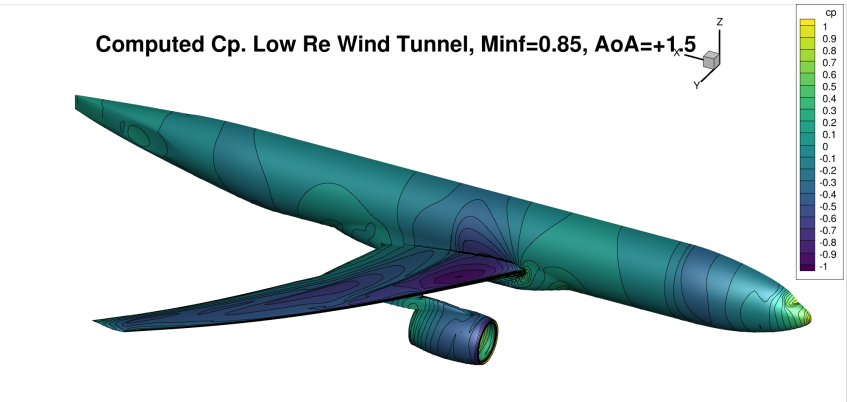
5.2 Numerical aspects

Reynolds-Averaged Navier-Stokes(RANS) with Spalart-Allmaras turbulence model simulations were carried out utilising the elsA solver, with a structured mesh shared by the Boeing company during the *Drag Prediction Workshop 6 (DPW6)* [24]. The chosen mesh, which is denoted WBPN3 in [24], contains 39.5 million points and has an average half wall-cell height $y+$ of 0.50. It is a Chimera mesh, consisting of partly overlapping meshes in order to accurately represent the complex geometry.
Regarding the numerical scheme, a finite volume method with low numerical dissipation

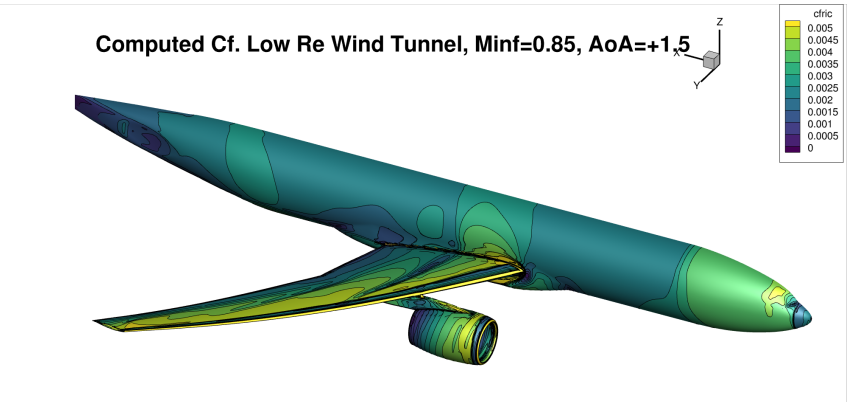
was utilised. Additionally, a two-level multigrid algorithm was employed.
To ensure consistency with low/moderate Re experiments that made use of rough adhesive strips to force turbulent transition, forced laminar zones of specific width were established on the surface of the front part of the wing, body and nacelle. Forced laminar zones work by imposing $\mu_t = 0$ in all the cells whose closest wall interface is in the forced laminar zone. This induces the transition to turbulence at the correct location with respect to the experiments.
The geometric details of this procedure were based on the experimental setup detailed in [25], where the adhesive strips were positioned at specific locations of a wind tunnel model.
To carry out the numerical simulations, the mesh was partitioned in 96 domains, and the computation of each domain was relegated to a specific core. In total, 96 cores from 3 Intel Xeon "Skylake - 6152" nodes were utilised for each simulation for a computing time of around *8h30min*.
The personal contributions of the author to this project consist in managing the computations, post-processing the elsA outputs and producing the figures of interest for a third of the entire dataset (case $p_i = 100000Pa$). Besides, elementary aerodynamic controls on the adimensional force coefficients in the geometric frame of reference $C_x(AoA)$ and $C_z(AoA)$ were performed.

5.3 Visualisation of results

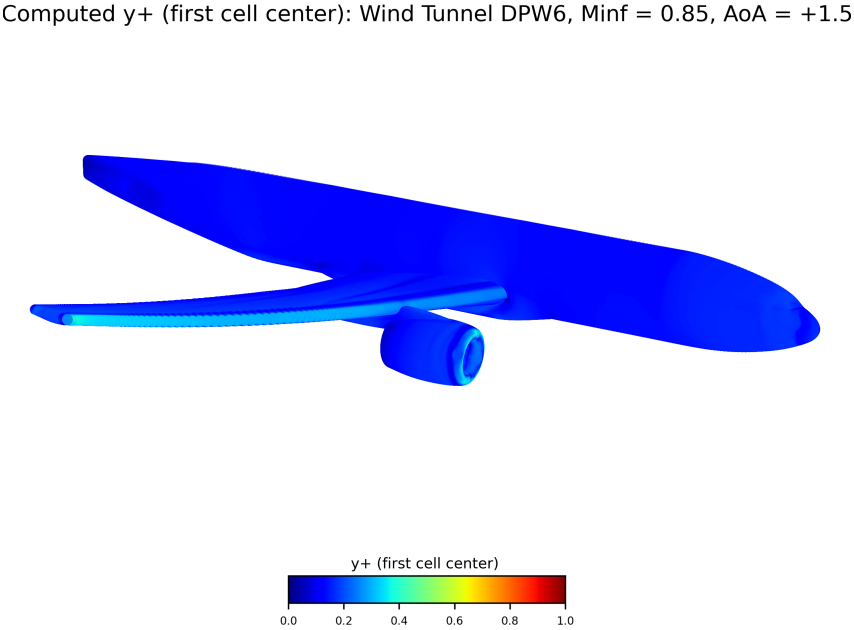
To visualise a portion of the results, the cruise condition case of $(M_\infty, AoA) = (0.85, 1.5°)$ is taken into consideration. Specifically, pressure coefficient C_p, skin friction coefficient C_f and $y+$ fields are shown in the figures below for each value of the stagnation pressure p_i: Figure 16 presents the case of $p_i = 100000Pa$, Figure 17 the case of $p_i = 200000Pa$ and Figure 18 the case of $p_i = 400000Pa$. The Re of these flows, based on the far-field quantities and the mean wing chord are respectively $Re = 2.5\,10^6$, $5.0\,10^6$ and $10.0\,10^6$.
When looking at the C_p distributions, an area of very low C_p can be observed on the upstream upper side of the wing, in line with the expectations for a civil aircraft such as CRM. This zone is terminated by a weak shock wave, which is also expected. Additionally, when looking at the C_f distributions, a thin strip presenting very low C_f can be identified at the front of the wing, body and nacelle. The presence of such strip is justified by the introduction of the forced laminar zones described in §5.1. Finally, the values of $y+$ appear to be in the acceptable range, with the exception of small areas in the case of $p_i = 400000Pa$ at the wing leading edge, close to the tip. Also expected from the simulations are the similarities of the C_p field when varying the Re, and the increasing C_f when the Re diminishes.



(a) C_p field.

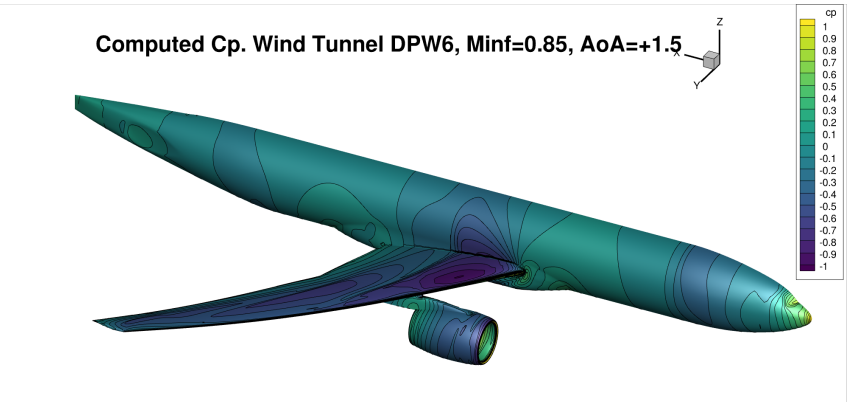


(b) C_f field.

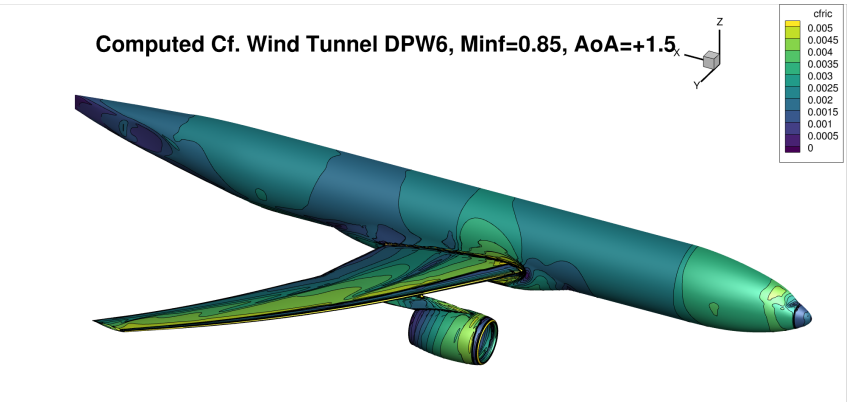


(c) $y+$ field.

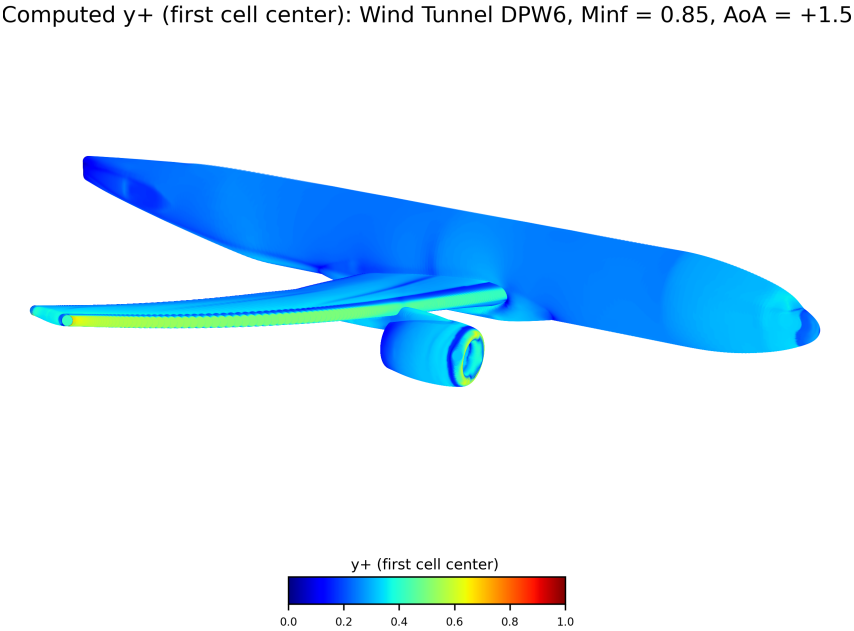
Figure 16: Resulting fields of pressure coefficient C_p, skin friction coefficient C_f and $y+$ for the case $p_i = 100000 Pa$, $M_\infty = 0.85$, $AoA = 1.5°$.



(a) C_p field.

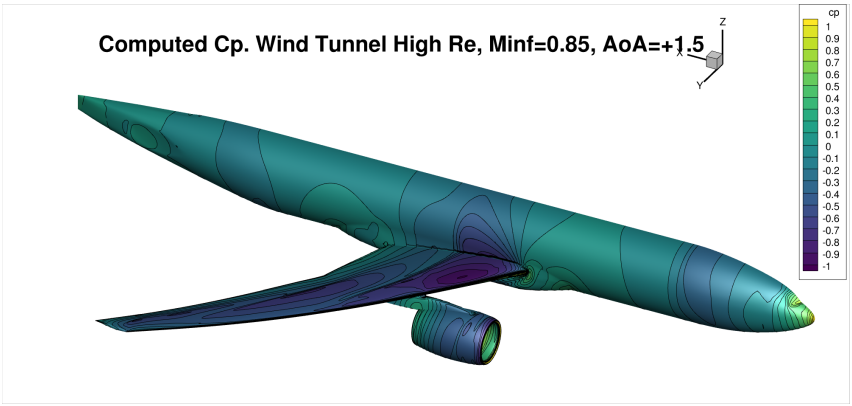


(b) C_f field.

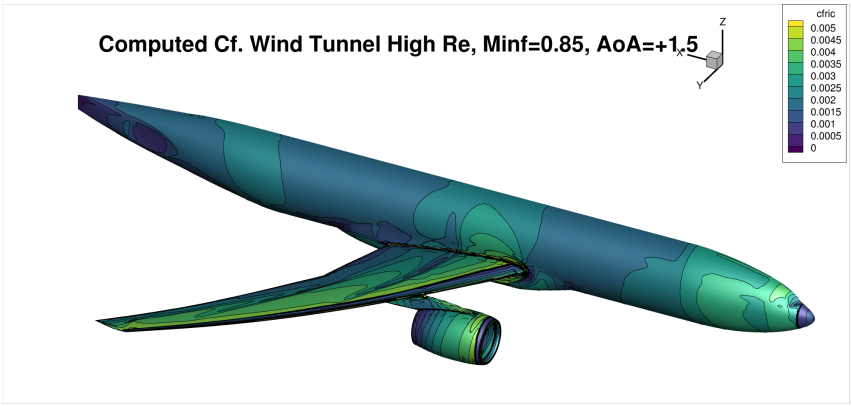


(c) $y+$ field.

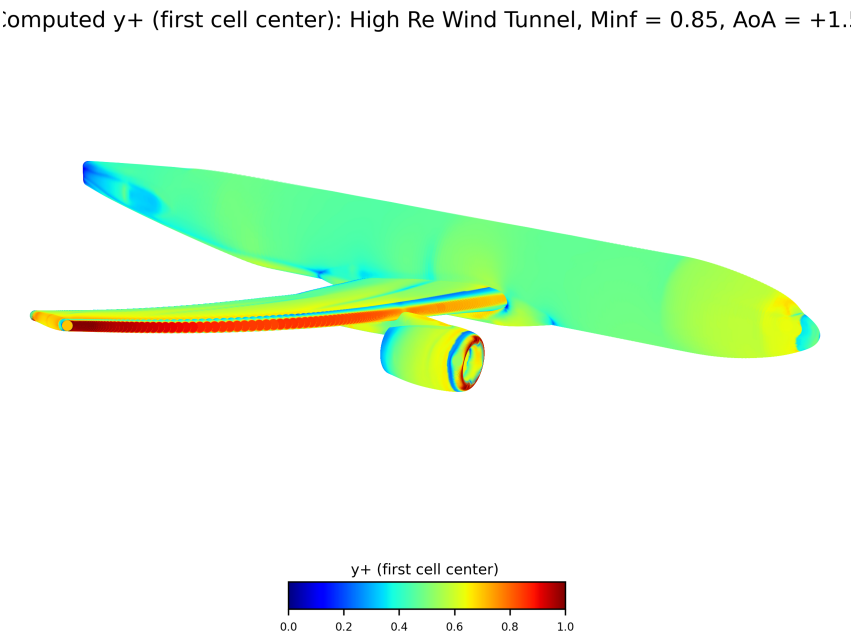
Figure 17: Resulting fields of pressure coefficient C_p, skin friction coefficient C_f and $y+$ for the case $p_i = 200000 Pa$, $M_\infty = 0.85$, $AoA = 1.5°$.



(a) C_p field.



(b) C_f field.



(c) $y+$ field.

Figure 18: Resulting fields of pressure coefficient C_p, skin friction coefficient C_f and $y+$ for the case $p_i = 400000 Pa$, $M_\infty = 0.85$, $AoA = 1.5°$.

6 CONCLUSIONS

For Part 2, the computation and postprocessing of the CFD simulations were successfully carried out. Tracing preliminary plots of the significant aerodynamic quantities suggests that the resulting dataset can provide high quality parietal fields for future Machine Learning tasks, including a continuation of the work presented in Part 1.

For Part 1, a large number of ROMs has been tested on a regression task based on the XRF1 model which was defined by the GARTEUR group AG60.
Firstly, k_{NN} interpolation appeares to be a very effective model in spite of its simplicity. Plotting absolute error distributions revealed that the error for values of M_∞ in the proximity of the cruise M_∞ is very localised. This last observation, along with the known limitations of predictions based on linear combinations in $\mathcal{Y}$ within the transonic regime, highlight Optimal Transportation methods, such as those discussed in [26], as a topic of interest for future research.
Studies on the different dimensionality reduction techniques revealed that performing regression from a $\mathcal{Z}$ space obtained by Isomap can be beneficial, but the improvements in performance can ultimately be matched by a carefully trained "generalist" MLP regression model operating from $\mathcal{P}$ space. Nevertheless, when looking at ease and speed of both training and parameter tuning, Isomap's performance is unmatched.
While Deep Learning methods perform comparably to traditional Machine Learning techniques in regression tasks, the same cannot be said for dimensionality reduction. In the data-scarce regime of this study, using models that enforce correct priors on the latent space structure, such as Isomap's global isometry, proves to be highly beneficial. General Deep Learning methods often underperform, sometimes struggling to recover meaningful parameterizations. In contrast, Isomap generates insightful low-dimensional representations that effectively capture a significant portion of the information present in the high-dimensional dataset $\mathbf{Y}$. This suggests promising applications of dimensionality reduction techniques in scenarios where the problem low-dimensional parameterisation is not as apparent as the (M_∞, AoA) case of this study, and has to be discovered.
Finally, the results clearly identify MLP pointwise as the best performing model. The very competitive performance is believed to be linked with the integration of other physical aspects of the C_p generating process, such as point coordinates. Future work will focus on further analysing pointwise methods, involving not only local coordinates but also local normal vectors in the inputs of the network. Mesh information and local exchanges between neighboring points (mimicking the dependencies of an explicit scheme) may even be included in further steps.
The framework of Graph Neural Networks (GNNs) offers a valuable methodology to process mesh related information, specifically in the case of GNNs introducing complex node interactions, such as the Message Passing scheme [27] and its generalisation in the Encoder-Processor-Decoder architecture [28]. At the same time, GNNs have stricter requirements on their input data, which has to be provided with connectivity information and not as a simple list of points. Very promising results on the application of GNNs to fluid dynamics [29] and specifically pressure field predictions [30] have already appeared in the literature. Particularly, the generalisation of the convolutional layer (one of the cornerstones of Deep Learning's success in image recognition) to mesh geometries is of

particular interest to the scientific community. Different alternatives exist [31] [32], all presenting a tradeoff between expressivity and scalability, some of which have already been applied to the task of pressure field prediction [33].

References

[1] Andrès, E. et al. *AD-AG60 Final Report.* Machine learning and data-driven approaches for aerodynamic analysis and uncertainty quantification. 2024.

[2] M. Carini, M., Blondeau, C., Fabbiane, N., Méheut, M., Abu-Zurayk, M., Feldwisch, J., Ilic, C., Merle, A. *Towards industrial aero-structural aircraft optimization via coupled-adjoint derivatives.* In AIAA paper series. AIAA paper 2021-3074. 2021.

[3] Bettebghor, D., Bartoli, N., Lefebvre, S. Machine Learning dans un contexte aéronatique. EUROSAE-ONERA Course. 2021.

[4] Goodfellow, I., Bengio, Y., Courville, A. *Deep learning.* MIT Pres. http://www.deeplearningbook.org. 2016.

[5] Rumelhart, D.E., Hinton, G.E. and Williams, R.J. Learning representations by back-propagating errors. *nature*, **323(6088)**, pp.533-536. 1986.

[6] Sirovich, L. Turbulence and the dynamics of coherent structures. I. Coherent structures. *Quarterly of applied mathematics*, **45**(3):561–571. 1987.

[7] Tenenbaum, J.B., Silva, V.D. and Langford, J.C. A global geometric framework for nonlinear dimensionality reduction. *science*, **290(5500)**: 2319-2323. 2000.

[8] Torgerson, W.S. Multidimensional scaling: I. Theory and method. *Psychometrika*, **17(4)**: 401-419. 1952.

[9] Saul, L.K. and Roweis, S.T. Think globally, fit locally: unsupervised learning of low dimensional manifolds. 2003.

[10] Belkin, M., Niyogi, P. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, **15(6)**: 1373-1396. 2003.

[11] Coifman, R. R., and Lafon, S. Diffusion maps. *Applied and computational harmonic analysis*, **21(1)**: 5-30. 2006.

[12] Le Cun, Y., and Fogelman-Soulié, F. Modèles connexionnistes de l'apprentissage. *Intellectica*, **2(1)**: 114-143. 1987.

[13] Kingma, D. P., and Welling, M. Auto-encoding variational Bayes. arXiv preprint arXiv:1312.6114. 2013.

[14] Kingma, D.P. and Welling, M. An introduction to variational autoencoders. *Foundations and Trends in Machine Learning*, **12(4)**, pp.307-392. 2019.

[15] Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R.R. and Smola, A.J. Deep sets. *Advances in neural information processing systems*, 30. 2017.

[16] Godbole, V., Dahl, G. E., Gilmer, J., Shallue, C. J., and Nado, Z. Deep Learning Tuning Playbook. *Version 1.0.* 2023. Available: `http://github.com/google-research/tuning_playbook`.

[17] Shallue, C.J., Lee, J., Antognini, J., Sohl-Dickstein, J., Frostig, R. and Dahl, G.E., Measuring the effects of data parallelism on neural network training. *Journal of Machine Learning Research*, **20(112)**, pp.1-49. 2019.

[18] Akiba, T., Sano, S., Yanase, T., Ohta, T., and Koyama, M. Optuna: A next-generation hyperparameter optimization framework. *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining* (pp. 2623-2631). 2019.

[19] Pedregosa, F., Varoquaux, G., Gramfort, A. Scikit-learn: Machine learning in Python. *the Journal of machine Learning research*, 12, 2825-2830. 2011.

[20] Paszke, A., Gross, S., Massa, F. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32. 2019.

[21] Saves, P., Lafage, R., Bartoli, N., Y. Diouane, Y., Bussemaker, J., Lefebvre, T., Hwang, T., Morlier, J., Martins, J.J.R.A. SMT 2.0: A Surrogate Modeling Toolbox with a focus on Hierarchical and Mixed Variables Gaussian Processes. *Advances in Engineering Sofware*, **188**:103571. 2024.

[22] Cambier, L., Heib, S. and Plot, S. The Onera elsA CFD software: input from research and feedback from industry. *Mechanics & Industry*, **14(3)**, pp.159-174. 2013.

[23] Vassberg, J., Dehaan, M., Rivers, M., and Wahls, R. Development of a common research model for applied CFD validation studies. In *26th AIAA applied aerodynamics conference* (p. 6919). 2008.

[24] Hue, D., Chanzy, Q., and Landier, S. DPW-6: drag analyses and increments using different geometries of the common research model airliner. *Journal of Aircraft*, **55(4)**, 1509-1521. 2018.

[25] Cartieri, A., Hue, D., Chanzy, Q., and Atinault, O. Experimental investigations on common research model at ONERA-S1MA–Drag prediction workshop numerical results. *Journal of Aircraft*, **55(4)**, 1491-1508. 2018.

[26] Iollo, A., Taddei, T. Mapping of coherent structures in parameterized flows by learning optimal transportation with Gaussian models. *Journal of Computational Physics* **471**:11167. 2022.

[27] Gilmer, J., Schoenholz, S.S., Riley, P.F., Vinyals, O. and Dahl, G.E. Neural message passing for quantum chemistry. In *International conference on machine learning* (pp. 1263-1272). PMLR. 2017.

[28] Battaglia, P.W., Hamrick, J.B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V., Malinowski, M., Tacchetti, A., Raposo, D., Santoro, A., Faulkner, R. and Gulcehre, C. Relational inductive biases, deep learning, and graph networks. arXiv preprint arXiv:1806.01261. 2018.

[29] Lam, R., Sanchez-Gonzalez, A., Willson, M., Wirnsberger, P., Fortunato, M., Alet, F., Ravuri, S., Ewalds, T., Eaton-Rosen, Z., Hu, W. and Merose, A. GraphCast: Learning skillful medium-range global weather forecasting. arXiv preprint arXiv:2212.12794. 2022.

[30] Hines, D. and Bekemeyer, P. Graph neural networks for the prediction of aircraft surface pressure distributions. *Aerospace Science and Technology*, 137, p.108268. 2023.

[31] Masci, J., Boscaini, D., Bronstein, M. and Vandergheynst, P. Geodesic convolutional neural networks on riemannian manifolds. In *Proceedings of the IEEE international conference on computer vision workshops* (pp. 37-45). 2015.

[32] De Haan, P., Weiler, M., Cohen, T. and Welling, M. Gauge equivariant mesh CNNs: Anisotropic convolutions on geometric graphs. arXiv preprint arXiv:2003.05425. 2020.

[33] Baque, P., Remelli, E., Fleuret, F. and Fua, P. Geodesic convolutional shape optimization. In *International Conference on Machine Learning* (pp. 472-481). PMLR. 2018.

ONERA
THE FRENCH AEROSPACE LAB

www.onera.fr

PROPOSITION DE STAGE EN COURS D'ETUDES

Référence : **DAAA-2024-15** *(à rappeler dans toute correspondance)*	Lieu : Châtillon
Département/Dir./Serv. : DAAA	Tél. : 01 46 73 41 84
Responsable(s) du stage : Jacques Peter (DEFI), Sébastien Heib (CLEF)	Email. : jacques.peter@onera.fr

DESCRIPTION DU STAGE

Thématique(s) : Data analysis

Type de stage : ☒ Fin d'études bac+5 ☐ Master 2 ☐ Bac+2 à bac+4 ☐ Autres

Intitulé : Surrogate modeling of external aerodynamic simulations using machine learning and POD

Sujet : Computational Fluid Dynamics (CFD) is a mandatory but, for some problems, expensive tool for aerodynamic analysis and design. In particular, when considering a large space of flow conditions or design variables (input variables) or a very fine grid flow discretization (output variables), Surrogate Modeling or Reduced Order Model (ROM) from a limited number of CFD simulations may be useful to efficiently approximate the flow of interest.

This field has been changed by Machine-Learning methods and its large frameworks (PyTorch, Scikit Learn…) that have popularized recent specific techniques. In particular, it is now possible to look for a direct functional relationship between the input parameters and the flow on the solid body using Convolutional Neural Networks (CNN) [1]. The first part of the internship will be precisely devoted to the application of this method to a simple wing flow and a recently built data-base on a wing-body-pylon-nacelle configuration [2]. For both cases, the input parameters will be the angle of attack, the far-field Mach number and the Reynolds number.

An older method consists in extracting Proper-Orthogonal-Decomposition (POD) modes from the sample flows and defining the coefficients of the modes as classical Machine Learning (ML) surrogates depending on the input parameters [3,4]. In the second part of the internship, Kriging-POD will be studied and applied to the same two cases comparing the accuracy of the resulting surrogates.

If time permits, the trainee will study one more method using a particular type of neural network called autocoder that allows dimension reduction : here a CNN with decreasing layer sizes (encoder part) then increasing layer sizes (decoder part) is built with the goal to transform the calculated wall distributions into themselves. After the full network has been tuned to the data, the inputs of the decoder are built as functions of the problem input variables (in our case, the flow conditions).

[1] A. Guorges, R. Castellanos, J. Bowen, E. Andrès. A comparison of machine learning methods for pressure coefficient prediction of an aeronautical configuration. ECCOMAS 2022. Oslo. 2022.

[2] David Hue, Quentin Chanzy, Sam Landier. DPW-6: Drag Analyses and Increments Using Different Geometries of the Common Research Model Airliner. Journal of Aircraft, 2017.

[3] R. Dupuis, JC. Jouhaud, P. Sagaut. Surrogate modelling of aerodynamic simulations for multiple operating conditions using machine learning. AIAA Journal 56 (9). 2018.

[4] F. Sun, W-Y Su, M.Y. Wang, R.-J. Wang. RBF-POD reduced-order modeling of flow field in the curved shock compression inlet. Acta Astronautica. 185. 2021.

Est-il possible d'envisager un travail en binôme ? **Non**

Méthodes à mettre en oeuvre :

☒ Recherche théorique ☐ Travail de synthèse

☒ Recherche appliquée	☒ Travail de documentation
☐ Recherche expérimentale	☐ Participation à une réalisation
Possibilité de prolongation en thèse :	**Non**
Durée du stage : Minimum : 5 months	Maximum : 6 months
Période souhaitée :	

PROFIL DU STAGIAIRE

Connaissances et niveau requis : - Mathématiques Appliquées - Aérodynamique	Ecoles ou établissements souhaités : Master Recherche et/ou Ecole d'ingénieur