The 9th European Congress on Computational Methods in Applied Sciences and Engineering
ECCOMAS Congress 2024
3–7 June 2024, Lisboa, Portugal

COMPARISON OF DATA-DRIVEN METHODS FOR THE PREDICTION OF FLOWS ABOUT A WING

MATTEO LIPPI[1] AND JACQUES E.V. PETER[1]

[1] DAAA, ONERA, Institut Polytechnique de Paris, 92320, Châtillon, France
e-mail: peter@onera, www.onera.fr

Key words: Computational Fluid Dynamics, Machine learning, Regression, Aircraft

Summary. Reduced Order Models are applied to predict pressure distributions on a civil aircraft, at various combinations of Mach number and angle of attack. The efficiency of a pointwise method (computing individual Cp values from the wall point location and the flow conditions) and several modewise methods (computing wall Cp distributions from the flow conditions) including POD, IsoMap, Diffusion Map and Autoencoder, is assessed and discussed.

1 INTRODUCTION

ONERA's activities on parametrized CFD problems date back as far as the mid-80's when local shape optimization of airfoils were performed using finite-difference gradients [1, 2]. Since that time, of course, local shape optimizations were done based on (discrete) adjoint gradients [3, 4] regularly considering tens and hundreds of design parameters for 2D and 3D complex configurations [5, 6]. Fundamental properties of the adjoint method were also studied – [7, 8, 10, 11] – as well as global optimisation algorithms – [9] and references therein.
Besides, since the mid-2000's, Uncertainty Quantification problems have been considered, starting with basic tests and now dealing with realistic manufacturing tolerances / variations of operating conditions for performance assessment. Currently, polynomial chaos defined by (possibly gradient-enhanced) collocation techniques is the preferred method and its evaluation of standard deviation will soon be involved in robust optimizations [12, 13].
Since the internal project PR Chyne (2014-2017) and the EG67/AG60 GARTEUR initiative (2019-2023) [15], Reduced Order Models (ROM) of industrial CFD/EFD outputs as function of flow conditions, is a further topic of interest in the field of parametrized fluid dynamics. The current study focuses on the behavior and performance of a series of ROM for the reconstruction of parietal flows about the XRF1 civil aircraft configuration [14].

2 DATASET

The dataset for the comparison regression methods is one of those provided by Airbus-M in the framework of the AG60 Garteur group [15]. Various wing-body CFD calculations were performed by Airbus-M with the DLR TAU code for a XRF1 model in a wind tunnel (XRF1 configuration [14] is close to a long range wide body Airbus aircraft). Whatever the detailed geometric configuration (with or without vertical tail plane, horizontal tail plan or sting), the final provided data were Cp coefficients on the wing of the aircraft. The three provided sets of calculation outputs have been divided by INTA in four couples of (learning, testing) data in order to define as many regression exercises [15]. For the current study, we retain a set with

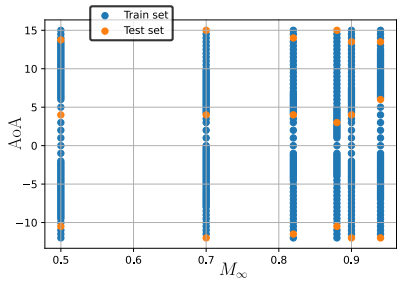

Figure 1: XRF1 flow conditions. Training and testing sets

consistent Reynolds numbers, varying (M_∞, AoA) and with same location of the data points on the wing for all flow conditions. More precisely $N = 112926$ Cp values are available for $n_p = 426$ flow conditions. Those have been divided in n_t=408 conditions for training and n_v=18 for testing the accuracy of the various regressors.
For sake of generality, in the following, one (M_∞, AoA) flow condition is denoted $\mathbf{p}$ (parameters of the regression) and the parameter space is denoted $\mathcal{P}$. A Cp value is denoted y, a Cp distribution is denoted $\mathbf{y}$ (vector of size N) and the complete set of training Cp distributions is denoted $\mathbf{Y}$ (matrix $N \times n_t$). The coordinates of individual wall points appear in pointwise regressors; they are denoted $\mathbf{c}$.
As the data generating process of $\mathbf{y}$ is governed solely by the $\mathbf{p}$ 2-dimensional variable, we note that all the N-dimensional data points $\mathbf{y}_i$ must lie on a 2-dimensional manifold $\mathcal{Y}$ embedded in $\mathbb{R}^N$.

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). Unfortunately $N \times n_t$ is several orders of magnitude too high to search for pointwise regressors defining y as function of $(\mathbf{p}, \mathbf{c})$ (the linear system to solve would be of intractable size). Defining N classical (pointwise) surrogates for n_t data would be feasible although very expensive, but the surrogates for neighboring points may not be consistent with each other. For these reasons, we consider two other basic 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,

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

Equation (1) formalises the methodology, denoting the indexes of the k nearest neighbours of $\mathbf{p}^*$ by $\mathcal{N}(\mathbf{p}^*)$, and the inverse of $||\mathbf{p}_i - \mathbf{p}^*||$ (the Euclidean distance between $\mathbf{p}^*$ and one of the

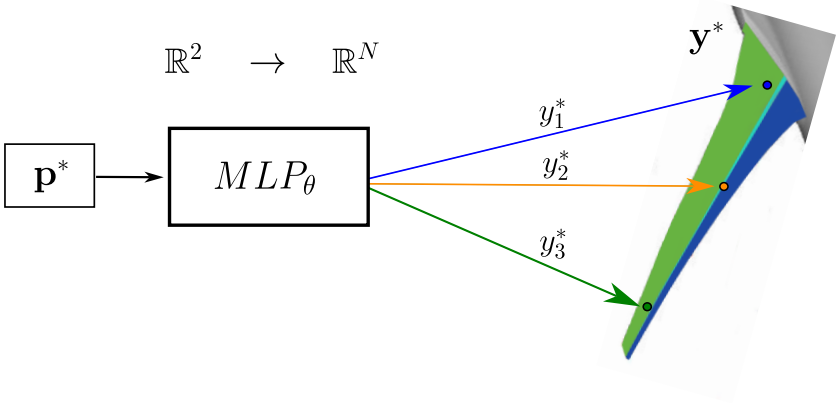

Figure 2: MLP Modewise model. Flow conditions $\mathbf{p}^*$ are mapped to their corresponding pressure distributions $\mathbf{y}^*$.

nearest points $\mathbf{p}_i$ of the training set) by w_i . This method does not involve any analysis of the output data 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 $\mathbf{p}$.

3.1.2 Reference method 2: MLP modewise

A second simple solution to the approximation 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

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

and learning the values of the inner parameters θ that result in the best approximation of f for the known $(\mathbf{p}_i, \mathbf{y}_i)$ data points [16]. More precisely, a MLP is defined by composing affine transformations $\mathbf{x}_l = \mathbf{W}\mathbf{x}_{l-1} + \mathbf{b}$ of the previous layer data with simple nonlinear functions such as *tanh*. Repeating this building block multiple times (hence the name "Deep Learning") yields the MLP architecture, where the aforementioned inner parameters θ 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 a loss function is applied.
At this stage, a MLP was trained to map the $\mathcal{Y}$ pressure distribution space from the $\mathcal{P}$ parameter space. This involves a very large number of weights between the last inner layer and the $\mathbf{y}$ output layer but is sustainable with current CPUs. A visual representation of the above model is presented in Figure 2. (See besides §3.5 for a pointwise MLP.)

3.2 Dimensionality reduction coupled with regression

A different approach consists in initially compressing the high dimensional pressure distributions $\mathbf{y}_i \in \mathbb{R}^N$ into a low dimensional representation $\mathbf{z}_i \in \mathbb{R}^r$ $(r << N)$, 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 an interpolation $\mathbf{z}^* = f(\mathbf{p}^*)$ with regular methods (RBF, GP...) becomes feasible. After interpolation, the data is mapped back to the high dimension

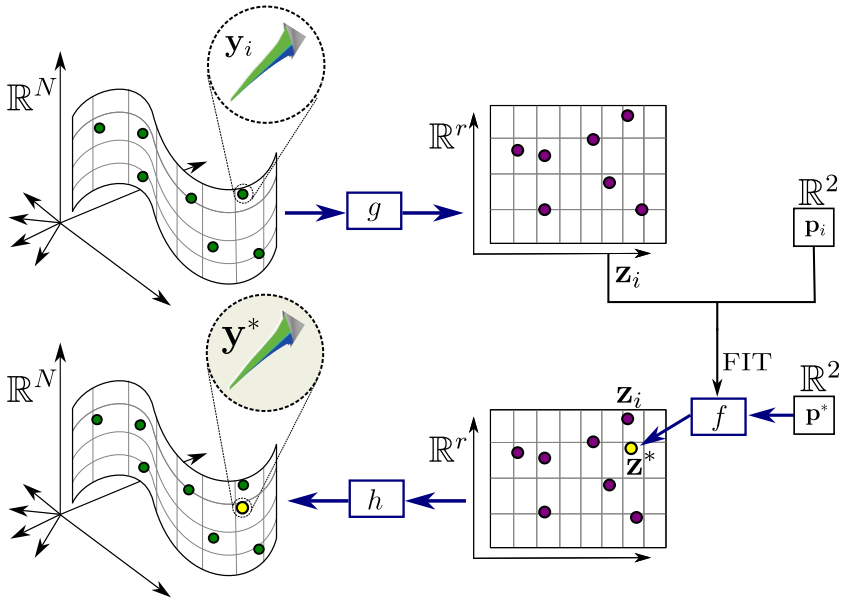

Figure 3: General principle of ROM involving dimensionality reduction.

using a reverse mapping function $h(\mathbf{z}^*)$. The entire modelling process is summarized in Figure 3.

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 $h_{int}(\mathbf{z}^*) : \mathbb{R}^r \rightarrow \mathbb{R}^N$ may seem redundant, but it can still be beneficial to perform dimensionality reduction: 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.

3.3 Linear dimensionality reduction (Proper Orthogonal Decomposition) plus Interpolation

Proper Orthogonal Decomposition (POD) [17] is a standard dimensionality reduction technique in fluid mechanics. POD firstly performs a Singular Value Decomposition (SVD) of the output matrix $\mathbf{Y} \in \mathbb{R}^{N \times n_t}$:

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

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 $\mathbf{Y}$.

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

where σ_k represents a diagonal element of Σ_r, $\mathbf{u}_k$ represents a column vector of $\mathbf{U}_r$, and $\mathbf{z}_k^t$ represents a row vector of $\mathbf{Z}_r^T$. 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 data $\mathbf{Y}$; 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 hyperplane that is optimal in the Frobenius norm sense, meaning that the reconstruction error $\|\mathbf{Y} - \mathbf{Y}_r\|_F$ 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 $\mathbf{y}_i$ on the r-dimensional plane.

Following the blueprint of Figure 3, 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).

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

POD is a powerful technique that yields an exact (lossless) embedding when the $\{\mathbf{y}_i\}$ lie in a r-dimensional vector space. At the same time, its weakness lies in the linear nature of the projection it applies. This does not usually pose a problem in the subsonic regime, but, in the transonic regime, when strong nonlinearities appear, $\mathcal{Y}$ 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 [18], which can be intuitively understood as an attempt to isometrically "unwrap" twisted manifolds. More specifically, the Isomap algorithm consists of three steps:

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) [19] 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 by

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

where $\mathbf{H} = \mathbf{I}_{n_t} - {n_t}^{-1}\mathbf{J}_{n_t}$ is a centering matrix, $\mathbf{J}_{n_t}$ being a $n_t \times n_t$ matrix of ones, and $\odot$ 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 [20], 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:

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

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

   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 Theorema Egregium, exact isometric embedding of $\mathcal{Y}$ in dimension $r = 2$ exists only for manifolds presenting zero Gaussian curvature at every point. For any other Gaussian curvature case, 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, 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 [21], 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 for close corresponding points 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 datapoints uniformly distributed on the manifold surface. In the (frequent) case of non uniformly distributed datapoints, $\mathbf{L}$ is no longer a good discretisation of $\mathcal{L}$, and a generalisation of Laplacian Eigenmaps called Diffusion Maps [22] can be utilised to recover a correct discretisation. Diffusion maps define 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 datapoints' density, with $\alpha = 0$ simplifying to $\mathbf{L}$, and $\alpha = 1$ defining 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 3 is applied. Regarding the interpolation, the reverse mapping procedures presented in §3.4.1 for Isomap are used.

3.4.3 Autoencoder

An autoenconder (AE) [23] 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 $\mathbf{z}$.
The high complexity of the loss landscape of an AE makes the training process prone to learning suboptimal parameters. Additionally, as many low dimensional representations may exist, an AE may learn a specific $\mathbf{z} = g_{AE}(\mathbf{y})$ that is not optimal for latent space interpolation.
A Variational Autoencoder (VAE) [24] 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 $p_\theta(\mathbf{y}, \mathbf{z}) = p_\theta(\mathbf{y}|\mathbf{z})p(\mathbf{z})$, whose parameters θ are given by a MLP (equivalent to the decoder network of the AE architecture). VAE attempts to solve the intractable Bayesian inference problem of finding $p_\theta(\mathbf{z}|\mathbf{y})$ by approximating it with a probability distribution $q_\phi(\mathbf{z}|\mathbf{y})$, parametrised by a second MLP (equivalent to the encoder of the AE). This setup allows a VAE to effectively learn an approximation $q_\phi(\mathbf{z})$ of the probability distribution

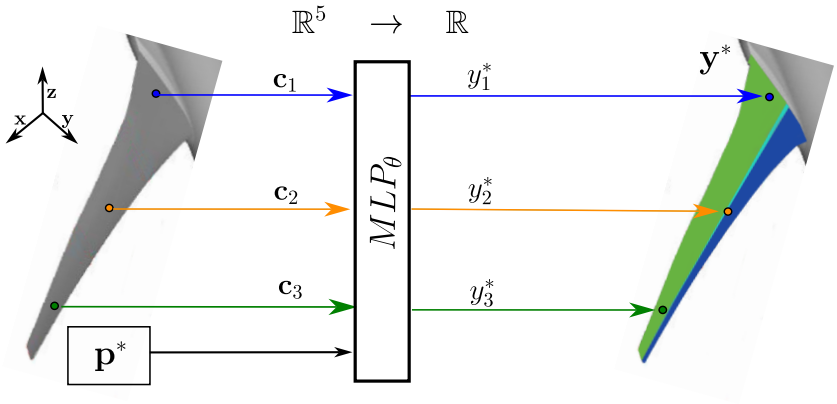

Figure 4: MLP Pointwise model. Flow conditions and point coordinates $(\mathbf{p}^*, \mathbf{c}_n^*)$ are mapped to the corresponding Cp value y_n^*.

of the latent variable $\mathbf{z}$, thus generalising the latent space concept of an AE. The characteristics of $q_\phi(\mathbf{z})$ 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 $\mathcal{Z}$.
In order to predict $\mathbf{y}^*$, the blueprint of Figure 3 is followed. 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 [25]. 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 $f_{MLP} : \mathbb{R}^5 \rightarrow \mathbb{R}$ 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 4.

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,

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

were calculated for the testing set ($n_v = 18$ distributions). Results are shown in Table 1.
The scikit-learn library [26] was utilised to carry out the implementation of all methods; the Pytorch library [27] was also leveraged for the implementation of all deep learning methods. (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 hyperparmeters shall be optimized using a reserved part of the training data –[15] §3.

- *(Control case) k_{NN} regression:* The number of neighbours was optimised by grid search and chosen to be $k_{\mathcal{P}} = 8$.
- *(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 not improve performances. The activation function was *tanh*. Adam optimiser was used to decrease a standard MSE loss with $L2$ weight regularisation (coefficient $\alpha = 10^{-4}/n_t$).
- *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 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 $r = 5$, $k_{\mathcal{Y}} = 14$ for the definition of $\mathbf{D}_G$ and $k_{\mathcal{Z}} = 10$ for the backmapping.
- *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 $r = 60$, $k_{\mathcal{Y}} = 200$ for the definition of $\mathbf{L}$, $k_{\mathcal{Z}} = 13$ for the backmapping.
- *Autoencoder and Variational Autoencoder:* The same architecture and hyperparameters were used. In both cases, a 3-hidden-layer encoder with layer sizes $(1000, 200, 50)$ 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 $r = 5$. 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. Both models were trained using early stopping. In the specific case of VAE, the $p_\theta(\mathbf{y}|\mathbf{z})$ and $q_\phi(\mathbf{z}|\mathbf{y})$ were chosen to be respectively

$\mathcal{N}(h_\theta(\mathbf{z}), \sigma^2\mathbf{I})$ and $\mathcal{N}(g_{\phi 1}(\mathbf{y}), g_{\phi 2}(\mathbf{y})\mathbf{I})$, where $g_{\phi 1}$ and $g_{\phi 2}$ 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 (no improvement with more layers or neurons). The same optimization and activation function were selected as for MLP modewise. As the training set for the method is 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 structure of $\mathcal{Y}$ being highly nonlinear. It is also noted that dimensionality reduction models with stricter inductive priors, 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

Basic Methods	**R^2 score**	**RedDim Methods**	**R^2 score**
(Control) k_{NN} Regression	0.865	POD	0.845
(Control) MLP Modewise	0.872	Isomap	0.876
MLP Pointwise	0.923	Diffusion Maps	0.857
		AE	0.861
		VAE	0.864

5 PERSPECTIVES

A large number of Reduced Order Models has been tested on an approximation exercise of GARTEUR group AG60 based on the XRF1 model. Considering the results, future work will focus on further analysing pointwise methods, involving not only the local coordinates but also the local normal vector 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. Although basic k_{NN} interpolation of pressure distributions from the parameter space performed surprisingly well, we suspect that all methods (with or without dimensionality reduction) involving a linear combination in $\mathcal{Y}$ can not perform excellently when

transonic flows are involved. Therefore, methods defining transformations of the state variables inside the fluid domain, like Optimal Transportation methods [28], will also be considered in the future.

REFERENCES

[1] Reneaux J. Méthode de définition de profils par optimisation numérique. *La Recherche Aerospatiale* **5**:303–321. 1984.

[2] Reneaux J., Thibert; J.J. The use of numerical optimization for airfoil design. AIAA paper series, Paper 85-5026. 1985.

[3] Peter, J., and Dwight., R.P. Numerical sensitivity analysis for aerodynamic optimization: a survey of approaches. *Computers and Fluids*, **39**:373–391, 2010.

[4] Peter, J., Renac, F., Dumont, A., Méheut, M. Discrete adjoint method for shape optimization and mesh adaptation in the elsA code. Status and Challenges. In Proceedings of the 50th 3AF Symposium on Applied Aerodynamics, Toulouse, 2015.

[5] Carrier, G., Destarac, D., Dumont, A., Meheut, M., Salah El Din, I., Peter, J., Ben Khelil, S., Brezillon, J., Pestana, M. Gradient-based aerodynamic optimization with the elsA software. AIAA paper series, Paper 2014-0568. 2014.

[6] Meheut, M., Destarac, D., Ben Khelil, S., Carrier, G., Dumont, A., Peter, J. Gradient-based single and multi-points aerodynamic optimizations with the elsA software. AIAA paper series, Paper 2015-0263. 2015.

[7] Peter, J., Nguyen-Dinh, M., Trontin, P. Goal-oriented mesh adaptation using total derivative of aerodynamic functions with respect to mesh coordinates – with application to Euler flows. *Computers and Fluids*, **66**:194–214, 2012.

[8] Todarello, G., Vonck, F. Bourasseau, S., Peter, J., Désidéri, J.-A. Finite-volume goal-oriented mesh-adaptation using functional derivative with respect to nodal coordinates. *Journal of Computational Physics*, **313**:799–819, 2016.6.

[9] 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.

[10] Peter, J., Renac, F., Labbé, C. Analysis of finite-volume discrete adjoint fields for two-dimensional compressible Euler flows. *Journal of Computational Physics* **449**:110811. 2022.

[11] Peter, J., Désidéri., J.A. Ordinary differential equations for the adjoint Euler equations. *Physics of Fluids* **34**:086113. 2022.

[12] Ghisu, T., Lopez, D., Seshadri, P., Shahpar, S. Gradient-enhanced Least-square Polynomial Chaos Expansions for Uncertainty Quantification and Robust Optimization. In AIAA Paper series. AIAA 2021-3073. 2021.

[13] Bennehard, Q., Dergham, G., Peter, J., Carini, M. UQ Analysis and Robust Optimization of a Transonic Airfoil for Open Rotor Design. Proceedings of ECCOMAS Congress 2024.

[14] 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.

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

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

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

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

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

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

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

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

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

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

[25] 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.

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

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

[28] 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.