Technische
Universität
Braunschweig

Master Thesis

Investigation of novel approaches for aerodynamic data fusion

by

Anjan Prasad Kundapura

Matriculation Nr.: 4997333

under the supervision of

Dr. Anna Bertram & Dr. Philipp Bekemeyer

Institute of Aerodynamics and Flow Technology

German Aerospace Center (DLR), Braunschweig

Examiner

Prof. Dr. Stefan Görtz

Institute of Aerodynamics and Flow Technology

German Aerospace Center (DLR), Braunschweig

December 16, 2022

Declaration

I hereby confirm that the present master thesis was composed by myself and that the work contained herein is my own. I also confirm that I have not used any aids other than those indicated. The present thesis has not been submitted to another university for the award of an academic degree in this form.

Anjan

Anjan Prasad Kundapura

Braunschweig, 16.12.2022

(Place, Date)

Abstract

Modern industrial applications require reliable and accurate aerodynamic data for design and optimization. This data is generally produced using CFD simulations and wind tunnel testing. Although these approaches offer significant individual benefits, they also have certain limitations. CFD fails to yield accurate solutions towards the edge of the envelope, whereas the wind tunnel experimental data offers data only at specific sensor locations. Data fusion techniques combine the individual strengths of these data sources to deliver accurate and reliable data. POD-based data fusion techniques like Gappy POD and regularized Gappy POD are well-established and widely used in various studies. These techniques compute the data fusion result via a least-square fit in the POD subspace. Recently shallow artificial neural networks have also been used in data fusion techniques to reconstruct the flow solution. This thesis proposes an alternative data fusion approach called Gappy ANN and compares it with Gappy POD. The idea of Gappy ANN is to replace the POD subspace with a solution space generated via a shallow artificial neural network. The advantage of this approach is that knowledge of the sensor positions can be directly considered when creating the solution space. This thesis demonstrates the performance and robustness of Gappy POD and Gappy ANN on an aerodynamic test case fusing high-quality experimental and numerical data. Gappy POD performs better in reconstructing the flow solution than Gappy ANN, showing only minimal errors. To improve their prediction accuracy, DEIM-based sensor placement strategies are applied to the POD reduced space and ANN solution space to obtain the new optimal sensor locations. Gappy POD and Gappy ANN solutions are recomputed using these new locations. The results for both experimental and DEIM-based indices are analyzed to demonstrate the significance of the DEIM-based algorithm for the Gappy approaches in improving prediction accuracy.

Contents

Nomenclature

Latin Symbols

H	Measurement matrix
M	Mach number
N	Size of the computational grid
P	Mask matrix
Re	Reynolds number
S	Vector of sensor measurements
U, V	Orthonormal matrices
Y	Set of Snapshots
W^i	Weight matrix in the respective ANN hidden layers
C_p	Pressure coefficients
b	Bias
d	Size of the feature space
f	Nonlinear activation function
r	Rank of the snapshot matrix
s	Size of S
x	X coordinates
l_j^i	Input at the neurons in their respective ANN hidden layer
n_1	Size of the first hidden layer
n_2	Size of the second hidden layer
u_i	POD basis vectors
w_j^i	ANN basis vectors at the neurons in their respective ANN hidden layer
y_i	Each sample with d feature space
$\hat{a}$	POD basis coefficients / ANN basis coefficients
$\hat{y}$	Approximated full flow solution

Greek Symbols

Σ	Diagonal matrix containing singular values
λ	Penalization parameter
γ	Slope in the kernel function
σ_i	Entries in the Diagonal matrix Σ
ξ	Location of the dominant POD mode / ANN basis coefficient
α	Angle of attack

Abbreviations

ANN	Artificial Neural Network

CFD	Computational Fluid Dynamics
DEIM	Discrete Empirical Interpolation Method
FCNN	Fully Connected Neural Network
MSE	Mean Squared Error
POD	Proper Orthogonal Decomposition
SNN	Shallow Neural Network
SVR	Support Vector Regression
WTT	Wind Tunnel Testing

1 Introduction

An aircraft's development entails acquiring aerodynamic data for different flight conditions. Essential parameters like lift coefficients or the pressure distribution across the Aircraft's surface are used in the multidisciplinary analysis, such as structural, performance, and load evaluations. Modern industrial applications typically generate this data through Computational Fluid Dynamics (CFD) [9] simulations. Nevertheless, due to their computational complicatedness in handling large-scale dynamical and control systems, modern mathematical models dealing with real-life industrial problems constantly pose challenges in applying numerical simulations. In some flight conditions, CFD data is supplemented by the wind tunnel testing results because CFD, as a standalone, cannot provide accurate results. However, this raises a discrepancy between the computational and experimental data as wind tunnel testing results cannot provide the data at each grid point of the discretized Aircraft due to the limited designated sensor locations compared to the CFD solution, which records the results at each grid location.

Given the high cost of CFD, the critical importance of research today is to uncover ways to lower computational time while retaining the high fidelity of the analysis. Higher-fidelity data typically comprise more detailed physics and are computationally expensive to evaluate than lower-fidelity data. Lower-fidelity data sources are generally more affordable, but the solution produced using these data neglects significant physical effects present within the higher-fidelity data. Hence Data fusion techniques combine the individual strengths of different data sources to provide consistent, accurate, helpful information and reliable data sets. Variable-fidelity surrogate models (VFM) are a popular data fusion strategy for scalar-valued quantities. VFM, also known as multi-fidelity modeling, refers to the exploitation of two or more data layers of variable fidelity to create an inexpensive emulator of a given high-order numerical model. The most popular method employed for VFM is a correction-based method. The correction is called the "bridge function." The high-fidelity data is then approximated by a surrogate model for the low-fidelity data via an additive, multiplicative, or hybrid correction [22].

In a study, [27], data from two computer codes of different fidelity were combined using a simple "additive bridge function" with the Kriging method. Gaussian process regression, also known as Kriging, is a method of interpolation based on the Gaussian process governed by prior covariances with an assumption that the given sample points are realizations of correlated random variables. It can handle highly nonlinear responses and features fast evaluation times, making it a famous method in various fields, likely to design and analyze computer experiments [37, 43], machine learning [35], and surrogate modeling [18]. Other well-known VFM techniques, such as Cokriging [23] and Hierarchiel Kriging [21], have also been successfully used to model scalar-valued quantities of interest in various engineering fields. These techniques are applied in aerodynamic applications for variable-fidelity modeling based on numerous sources of computational data [18, 24, 8], experimental data, and numerical data [29]. These two techniques directly extend Gaussian process regression towards a variable fidelity framework. This study also demonstrated a non-hierarchial approach for the fusion of scalar-value quantities in which a gaussian process model is built for every data source individually.

The surrogate modeling of vector-valued quantities based on the variable fidelity methodology was shown in [4, 7]. The surrogate model for high-fidelity data was built with the aerodynamic data from various computer simulations of different accuracy on the same computational grid. A common orthonormal basis was computed using the dimensionality reduction technique Proper Orthogonal Decomposition (POD) [34, 3]. Using a variable-fidelity surrogate model like Cokriging or Hierarchial Kriging, the scalar-valued basis coefficients of this POD basis are interpolated. In a recently proposed data fusion strategy for vector-valued quantities [36], experimental and numerical data to the same aerodynamic flow conditions and spatial grid were combined via a weighted sum in a Bayesian setting. However, introducing low and high-fidelity data on the same computational grid may result in disfavor as it prevents their application for the fusion of sparse experimental sensor data and high-dimensional numerical data. This disadvantage was overcome by the Gappy POD method, which combines POD with a least-square problem to reconstruct not completely known data. The key concept is that in a least-square sense within the subspace spanned by the POD modes, a solution can be found which minimizes the differences to reference data at a few discrete locations. Gappy POD was first proposed to reconstruct images, such as human faces, from partial data [17]. Later the Gappy POD technique was extended to aerodynamic applications in reconstructing incomplete aerodynamic data [10]. This study has shown that Gappy POD is an effective technique for reconstructing complete flow solutions from limited surface measurements for the case of steady aerodynamic flow around an airfoil. Subsequently, a regularized Gappy POD approach was introduced to avoid overfitting [19]. The authors fused CFD and experimental surface pressure data from a steady aerodynamic flow around the flap of a transport aircraft using their regularized Gappy POD approach. This methodology was later extended such that the fused aerodynamic surface data sums up the overall integral coefficients as measured by the wind tunnel balance in the study [32]. However, the Gappy POD approach is impacted by an assumption of linearity within the POD model and the consecutive least-squares problem. In addition, such data-driven approaches often have the least acceptance rate because it is hard to estimate the uncertainty in the predicted data. This issue was addressed in the study [6], where the authors proposed a Bayesian extension for Gappy POD. This proposal aims to solve the Gappy POD problem by employing Gaussian process regression considering nonlinear covariance functions.

In the study, [42] Gappy POD method is used to handle the unsteady flow reconstruction problems. Based on this procedure, a systematic approach for optimal sensor placement is developed. Study [40] used a heuristic technique to decide the sensor locations by placing them at spatial maxima and minima of each POD mode in the investigation of "feedback flow control on the wake of a circular cylinder." The study [44] used the Gappy POD to find efficient sensor placement in the adaptive sampling for regional ocean forecasting. DEIM (Discrete Empirical Interpolation Method) [12] is an extension of POD that aims to reconstruct the high-dimensional state with a low-dimensional data set representation by selecting interpolation points in the POD reduced space. Although it was initially developed as a variant of the dimensionality reduction technique, it is used extensively in DEIM-based sensor placement strategies in [30, 26, 13, 14, 15].

This thesis work aims to investigate the standard POD-based data fusion techniques and proposes a new approach called Gappy ANN, in which the POD-based solution space is replaced with a space generated via a shallow artificial neural network (ANN) and computes a data fusion result via a least-squares fit in this space. The proposed and standard methods' performance and robustness are exemplified by investigating a test case fusing high-quality experimental and numerical data. The investigation is then extended to the optimal sensor placement strategy based on the DEIM algorithm. Alternative to the POD-based DEIM, a new method is proposed by replacing the POD solution space with the solution space generated by ANN. The new locations obtained through the POD-based DEIM and ANN-based DEIM algorithms are used again in their respective Gappy data fusion approaches to check for any potential improvements in their predictions.

This thesis is structured as follows. A theoretical introduction to POD and the standard methods based on the POD, likely ordinary and regularized Gappy POD is provided in sections 2.1 and 2.2 followed by kernel-based regression methods to solve the Gappy least-square problem in section 2.3. An introduction to Shallow artificial neural networks and their application in flow reconstruction is discussed in section 2.4, followed by a theoretical introduction to the proposed Gappy ANN in section 2.5. The details of the DEIM-based algorithm are discussed in section 2.6. The results for the discussed data fusion approaches and their respective DEIM methods are then discussed in chapter 3. This thesis is closed by giving a conclusion and future scope in chapter 4.

2 Theoretical Foundations

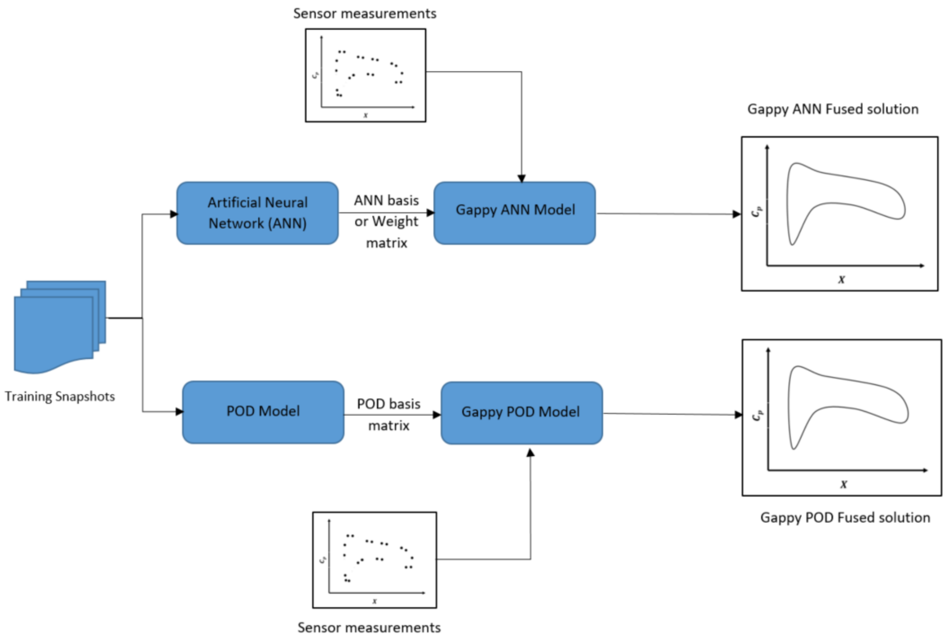


Figure 2.1: Workflow of data fusion approaches

Proper Orthogonal Decomposition (POD) is a linear dimensionality reduction technique based on Singular Value Decomposition (SVD) [41], often used to generate a low-rank, orthogonal basis that optimally represents a set of data. Instead of computing the desired quantity within the high-dimensional full-order space, it is determined within a low-dimensional subspace known as "Principal Subspace," spanned by the so-called POD basis, which is constructed using a finite number of sample values of the quantity of interest. The POD technique is summarized below.

Let us consider a set of N snapshots s.t. $Y = \{y_i | i = 1, ..., N\}$ with each sample y_i having d features i.e. $y_i = (y_{i,j})_{j=1,...,d}$. The samples are assumed to be centered around the mean without loss of generality. The resulting outcomes are stored in the snapshot matrix $Y \in \mathbb{R}^{N \times d}$. An SVD is performed on the snapshot matrix Y, which yields,

$$Y = U \Sigma V^T, \tag{2.1}$$

where $U = [u^1, ..., u^N] \in \mathbb{R}^{N \times N}$ and $V = [v^1, ..., v^d] \in \mathbb{R}^{d \times d}$ are orthonormal matrices s.t. $U^T U = U U^T = I_N$ and $V^T V = V V^T = I_d$. Σ is a diagonal matrix containing the singular values in descending order, i.e., $\sigma_1 \geq, ..., \geq \sigma_d \geq 0$. Considering r as the snapshot matrix rank, it can be inferred that only the first $r \leq d$ singular values are non-zero. The POD basis, i.e., $\{u^1, ..., u^r\}$, is then constituted by considering the r left singular vectors, which are also the first r columns of the matrix U. The data fusion technique Gappy POD utilizes these POD basis to compute the data fusion results via a least-squares fit as illustrated in Figure 2.1.

2.1 Gappy POD

A high-dimensional solution representation from a limited number of sensor measurements is possible if the dataset exhibit a low-dimensional structure. This attribute has been controlled for state estimation using either a tailored basis, such as POD or a general basis in which the signal is sparse. Both approaches are influenced by the fact that there exists a basis in which a high-dimensional state vector has a compressible or sparse representation [16].

Given a vector of experimental data $S \in \mathbb{R}^s$ with $s < N$ experimental sensors, a key idea of the Gappy POD is to interpret S as a vector $y \in \mathbb{R}^N$ from which only the components $y_{j^1}, ..., y_{j^s}$ where $j^1, ..., j^s \in \{1, ..., N\}$ are known. This can be accomplished by considering the relationship,

$$S = \begin{bmatrix} s_1 \\ \vdots \\ s_s \end{bmatrix} = \begin{bmatrix} y_{j^1} \\ \vdots \\ y_{j^s} \end{bmatrix} = P^T y, \tag{2.2}$$

where $P \in R^{N \times s}$ is a mask matrix. Assuming that y can be approximated using the POD subspace, let us find the first r POD basis coefficients $\hat{a} = (\hat{a}_1, ..., \hat{a}_r)^T$ with the help of the matrix of POD basis vectors $U_r = [u^1, ...u^r] \in \mathbb{R}^{N \times r}$ such that

$$y \approx \hat{y} = \sum_{j=1}^{r} \hat{a}_j u^j = U_r \hat{a}. \tag{2.3}$$

A standard way of approximating the solution is simply solving the least-square problem. The least-square problem defines the smallest L_2 error regarding the observed entries of the vector y from the basis coefficient vector $\hat{a} \in \mathbb{R}^r$ as

$$\hat{a} = \arg\min_{a} \left\| P^T U_r a - S \right\|_2^2 \tag{2.4}$$

Usually, $X = P^T U_r \in \mathbb{R}^{s \times r}$ has a complete column rank, so equation (2.4) has a unique solution as

$$\hat{a} = (X^T X)^{-1} X^T S. \tag{2.5}$$

Therefore the ordinary Gappy POD solution of the vector $\hat{y}$ can be obtained by substituting the basis coefficient $\hat{a}$ in equation (2.3).

2.2 Regularized Gappy POD

The ordinary Gappy POD method has a few shortcomings. Firstly, the least-squares problem formulated in equation (2.4) can be inaccurate due to the influence of vectors corresponding to small singular values. Hence it is favorable to impose some penalty on the basis coefficients $\hat{a}$ to reduce the variance by employing regularization terms. This will also help avoid overfitting, which is often necessary when dealing with data from different sources. Thus by scaling the basis vectors with their corresponding singular values, the influence of the less critical POD modes can be restricted.

Ridge regularization, also called Tikhonov regularization, is the most popular technique to reduce variance. The idea of this approach is to impose an $L2$ penalty on the basis coefficient vector a. From this technique, the corresponding Gappy POD problem reads,

$$\hat{a}_r = \arg\min_a \left\| P^T U_r a - S \right\|_2^2 + \lambda \left\| a \right\|_2^2, \tag{2.6}$$

where $\lambda > 0$ is the penalization parameter, which controls the regularization strength. The larger the value of λ, the higher the shrinkage towards 0. As seen in equation (2.3), if $X = P^T U_r$ has full column rank, the solution of the Gappy POD is unique. i.e.

$$\hat{a}_r = (X^T X + \lambda I)^{-1} X^T S. \tag{2.7}$$

So, substituting the value of $\hat{a}_r$ in the equation (2.3), the regularized Gappy POD approximation of the vector $\hat{y}$ is obtained as,

$$\hat{y} = \sum_{j=1}^{r} \hat{a}_j u^j = U_r \hat{a}_r. \tag{2.8}$$

2.3 Kernel based regression techniques to solve Gappy problem

As seen in section 2.1, approximating the solution of the Gappy POD involves solving a least-square problem. In this sense, one can understand the least-square problem of the Gappy POD as a standard regression problem that fits data by introducing a feature space. One way of solving this regression problem is by employing Kernel Ridge Regression (KRR). KRR combines ridge regression and linear least squares with L2-norm regularization with the famous Kernel Trick [33]. The kernel trick enables samples to operate in a high-dimensional, implicit feature space without ever entering one and calculating the coordinates of the data in that space by simply computing the inner product of all data pairs to reduce the computational complexities. The criteria to choose the kernel function $k(y_i, y_j)$ is given in the Mercer theorem [31]. From Mercer's theorem, $k(y_i, y_j)$, $y_i, y_j \in Y$ must be continuous, symmetric, and must have a positive semi-definite Gram matrix. Accordingly, the KRR learns a linear function in a feature space generated by the respective kernel function and the data. Another method to solve the linear regression problem is using Support Vector Regression (SVR) [11]. It is a supervised learning algorithm and non-parametric approach because it depends on kernel functions.

The accuracy of the data fusion technique also depends on the adequate selection of the kernel function $k(y_i, y_j)$ and their hyperparameters. Below is a list of kernel functions available from the literature [38]:

1. **Linear Kernel:** The linear kernel is the most straightforward function with the Euclidean inner product of $y^T \cdot y$. The linear kernel function is given as

$$k(y_i, y_j) = y_i^T y_j, \text{ where } y_i, y_j \in \mathbb{R}^d. \tag{2.9}$$

2. **Polynomial Kernel:** The Polynomial kernel function is given as

$$k(y_i, y_j) = (\gamma \cdot y_i^T y_j + c)^D \tag{2.10}$$

where the hyper-parameters γ is the slope, c is the constant term, and D is the polynomial degree $D \in \mathbb{N} = \{1, 2, ...\}$. If c = 0, the kernel is called homogeneous.

3. **Radial Basis Kernel:** The RBF kernel function is given as

$$k(y_i, y_j) = \exp\left(-\gamma \cdot \left\|y_i - y_j\right\|^2\right). \tag{2.11}$$

The parameter γ influences the outcome of the kernel and hence should be carefully chosen. If overestimated, the exponential will behave almost linearly, and the higher-dimensional projection will lose its non-linear power. On the other hand, if underestimated, the function will lack regularization [1].

4. **Sigmoid Kernel:** The Hyperbolic Tangent Kernel is known as the Sigmoid Kernel or the Multilayer Perceptron (MLP) kernel. The equation to compute the sigmoid kernel is given as

$$k(y_i, y_j) = \tanh\left(\gamma \cdot y_i^T y_j + c\right), \tag{2.12}$$

where γ is the slope, and c is the constant intercept term.

2.4 Shallow Neural Network in the application of flow reconstruction

Besides the standard POD-based data fusion methods, shallow learning techniques are also widely used [16]. An ANN may contain many hidden layers, but Shallow Neural Network (SNN) may consist of only 1 or 2, or none. The benefit of the SNN is that it allows faster training, less tuning, and more straightforward interpretation than deep networks. The linear last layer of the SNN delivers a supervised joint learning framework for the low dimensional approximation space of the flow field. Also, it maps the sensor measurements to this low-dimensional space. Consequently, this allows the approximation basis to be tailored to the associated measurements [16].

To construct such an SNN, first, a Fully Connected Neural Network (FCNN) is defined. For FCNN, every layer connects fully to the other layer, meaning every neuron in one layer connects to every neuron in the subsequent layer, as shown in Figure 2.2. The nodes in FCNN are commonly referred to as neurons, as FCNN will generally be referred to as neural networks. Given an input, it gives the output and delivers that output as an input to the following layer. Another significant benefit of FCNN is that they are "structure agnostic," i.e., no particular assumptions need to be made about the input if it consists of videos or images.

To represent the FCNN with l layers mathematically, let us consider the set of samples Y s.t. $Y = [y_1, ..., y_s] \in \mathbb{R}^s$ having data only at the s sensor locations. The output of the FCNN is defined as $\hat{Y}$ such that $\hat{Y} = [\hat{y}_1, ..., \hat{y}_N] \in \mathbb{R}^N$, where N represents the number of grid points in the computational grid. Generally the full output $\hat{Y}$ is computed as

$$\hat{Y} = f(W^l f(W^{l-1} ... f(W^1 Y))), \tag{2.13}$$

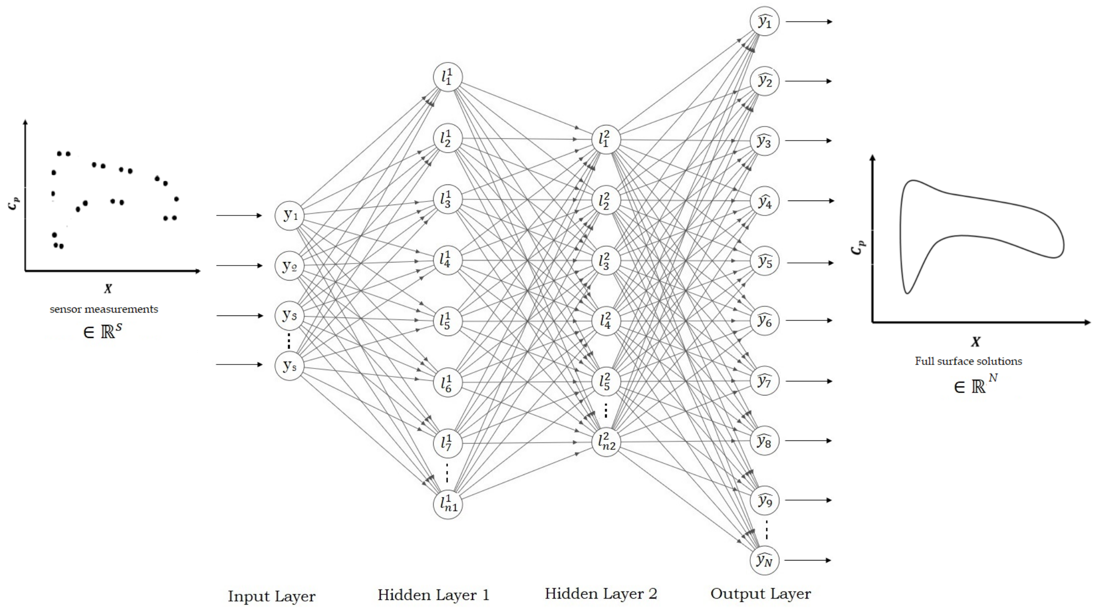


Figure 2.2: Architecture of the FCNN

where f is a coordinate-wise nonlinear activation function and W is a weight matrix, a learnable parameter.

The paradigm of simplicity guides architecture design. Indeed, the architecture should facilitate fast training and little tuning and present an intuitive interpretation. The chosen architecture in this thesis has only two hidden layers, which describe the mapping between the input sensor data S and the corresponding full surface solution as shown in Figure 2.2. The number of neurons at the input layer is selected per the number of features of the input sensor data $\mathbb{R}^s$. The size of the last hidden layer is chosen based on the POD model, i.e., the number of POD modes $\mathbb{R}^{n_2}$, although generally, the architecture of the network can have flexibility in the size of the neurons in the hidden layers. Based on the number of grid points in the full CFD solution $\mathbb{R}^N$, the same number of neurons are defined for the output layer.

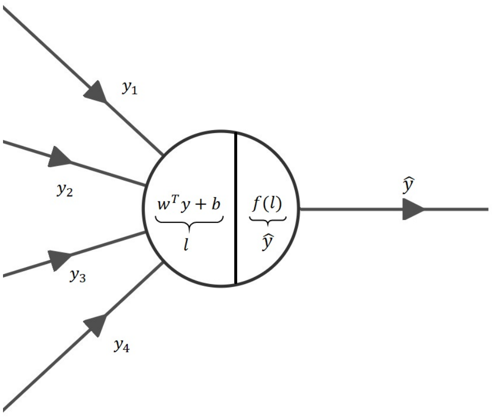


Figure 2.3: A Neuron in FCNN

The first and second hidden layers can be mathematically represented as

$$l^1 = f(W^1 y + b^1) \tag{2.14}$$

$$l^2 = f(W^2 l^1 + b^2), \tag{2.15}$$

where W denotes the dense weight matrix, b is a bias term and f an activation function to introduce non-linearity as illustrated in Figure 2.3. Thus the final linear output layer gets the form as

$$\hat{Y} = W^2 l^2 + b^2, \tag{2.16}$$

where $W^2 = [w_1^2, ..., w_{n_2}^2] \in \mathbb{R}^{N \times n_2}$ is weight matrix at the last hidden layer. The non-linearity to the hidden layer is introduced by employing an activation function Rectified Linear Unit (ReLU) [25] and a linear function to the output layer. It is among the most popular choice considering its advantageous properties. Notably, it is suitable for Sparse activation, better gradient propagation, efficient in computation, and scale-invariant [20]. The ReLU activation is defined as,

$$f(y) = y^+ = max(0, y), \tag{2.17}$$

where y is the input to neurons.

2.4.1 Optimizer and Loss function

The best way to train the network is to select the optimizer combined with the loss function and update the model parameter based on the response of the output of the loss function. This algorithm is called Backpropagation. The optimizer helps to build the model into its most accurate achievable form by updating the weights. The loss function acts as a navigator telling the optimizer when it is proceeding in the right or wrong direction.

Among the available optimizers in the literature, the SNN is trained using the Adamax optimizer [28], an extension to the Adaptive Movement Estimation (Adam) optimizer based on the gradient descent optimization algorithm. Adamax updates the weights inversely proportional to past gradients' scaled L_2 norm (squared). Hence it is based on the infinity norm (max) of past gradients. It is sometimes superior to adam optimizer, especially in models with embeddings. To understand how well the network has learned the feature data set, employing the loss function for the regression is essential. In other words, loss functions are a measurement of how good our model is in terms of predicting the desired outcome. In this thesis, the SNN model incorporates the most common loss function, Mean squared error (MSE). The loss is defined as the mean squared differences between target and predicted values or can be represented mathematically as,

$$MSE = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2, \tag{2.18}$$

where $\hat{y}_i$ is the predicted value of the model and y_i is the target value.

2.5 Gappy ANN

The core idea of the Gappy ANN is the same as that of the Gappy POD. However, unlike Gappy POD, the Gappy ANN uses the space generated by the SNN instead of the POD-based reduced space. The mathematical representation of the Gappy ANN is given below.

Let us consider a vector of sensor measurements $S \in \mathbb{R}^s$ with the missing snapshots for the full flow solution, implying $s < N$, only with the limited measurements on particular grid locations. Hence to interpret S as a full flow solution vector $y \in \mathbb{R}^N$ from which only the components $y_{j^1}, ..., y_{j^s}$ where $j^1, ..., j^s \in \{1, ..., N\}$ are known, then the relation can be shown as

$$S = \begin{bmatrix} s_1 \\ \vdots \\ s_s \end{bmatrix} = \begin{bmatrix} y_{j^1} \\ \vdots \\ y_{j^s} \end{bmatrix} = P^T y \tag{2.19}$$

where $P \in \mathbb{R}^{N \times s}$ is a mask matrix. Given that a well-trained weight matrix $W^2 \in \mathbb{R}^{N \times n_2}$ from the final hidden layer of the SNN is available, the solution $\hat{y}$ can be approximated as below

$$\hat{y} = W^2 \hat{a}, \tag{2.20}$$

where $\hat{a}$ in this sense is considered as ANN basis coefficients. Solving a least-square problem will be a standard way of approximating the solution. The least-square problem defines the smallest L_2 error regarding the observed entries of the vector y from the basis coefficient vector $\hat{a} \in \mathbb{R}^{n_2}$ as

$$\hat{a} = \arg\min_a \left\| P^T W^2 a - S \right\|_2^2. \tag{2.21}$$

Therefore as seen in the Gappy POD approach, the solution of the vector $\hat{y}$ from the Gappy ANN approach can be approximated by solving the lease-square problem in equation (2.21).

2.6 DEIM based optimal sensor placement strategies

2.6.1 POD-based DEIM

DEIM utilizes greedy algorithms based on pivoted factorizations of the dominant POD modes to efficiently select a nearly optimal collection of interpolation points. The standard POD methods are combined with their DEIM-based reduced order models to deliver a DEIM-based optimal sensor placement strategy. Using the DEIM interpolation points, the low-dimensional representation of high-dimensional data is given as

$$\hat{y}_r = Hy, \tag{2.22}$$

where $H \in \mathbb{R}^{r \times N}$ is the measurement matrix carrying the location information of the dominant modes and $\hat{y}_r$ is the low-dimensional representation of the data y. To update H with the location information, consider the POD basis matrix U retaining r left singular vector such that $U_r = [u^1, ...u^r] \in \mathbb{R}^{N \times r}$. The first index is computed as an initial step by capturing the dominant mode in the first POD basis vector.

$$[\rho, \xi] = max(u^1), \tag{2.23}$$

where $[\rho, \xi]$ denotes the dominant mode's value and location, respectively. Once the measurement matrix H is updated with this information, the DEIM algorithm iterates across the remaining basis vectors to find the location of the subsequent dominant modes as shown in algorithm 1.

Algorithm 1 POD-based DEIM algorithm

```
Require: $H, U_r, [u^1, ..., u^r]$
  Initialise $OptIndices_1 = \xi$
  for $i \leftarrow 2$ to $r$ do
      $C_i \leftarrow H_i^T u^{i+1} (H_i^T u^i)^{-1}$      ▷ Projection of next location to see the measurement
      $Res_{i+1} \leftarrow u^{i+1} - u^i C_i$      ▷ Find error (Residual)
      $[\rho, \xi_i] \leftarrow max(Res_{i+1})$      ▷ Check for dominant POD mode
      $H_{i+1} \leftarrow [H_i \xi_i]$      ▷ Update the $H$ matrix
      $OptIndices_i \leftarrow \xi_i$      ▷ Append the index location
  end for
```

The index list in the *OptIndices* will be in the decreasing order of the locations of the dominant modes, meaning the full-flow solution can be easily approximated with the help of the first few indices, irrespective of the dimension r. This algorithm is more effective in the optimal sensor placement strategy as one can calculate the complete flow information with the limited sensors placed in the prescribed grid location. Gappy POD predictions are taken again using the new optimal locations to analyze the performance of Gappy POD for POD-based DEIM locations.

2.6.2 ANN-based DEIM

Another objective of this thesis is to extend the algorithm of DEIM based on ANN basis to introduce a new optimal sensor placement strategy. This proposed method will replace the POD reduced space U_r with ANN solution space W^2 to get the new optimal sensor locations. Based on this notion, the data projection from high-dimensional to low-dimensional space is calculated as

$$\hat{y}_{n_2} = Hy, \tag{2.24}$$

where $H \in \mathbb{R}^{n_2 \times N}$ is the measurement matrix and $\hat{y}_{n_2}$ is the low-dimensional representation of the data y. The measurement matrix H will be updated using the ANN-based DEIM algorithm.

Let us consider again that a well-trained weight matrix $W^2 \in \mathbb{R}^{N \times n_2}$ from the last hidden layer with n_2 ANN basis vectors, i.e., $[w_1^2, w_2^2, ..., w_{n_2}^2]$ is available. As an initial step, the first index is found by capturing the dominant ANN basis coefficient in the first basis vector as

$$[\rho, \xi] = max(w_1^2), \tag{2.25}$$

where $[\rho, \xi]$ denotes the value and location of the dominant ANN basis coefficient, respectively. The measurement matrix H will be updated with this information. Once the initial step is complete, iteration will start to update the H for all the remaining ANN basis vectors, as illustrated in algorithm 2.

Algorithm 2 ANN-based DEIM algorithm

```
Require: $H, W^2, [w_2^2, ..., w_{n_2}^2]$
  Initialise $OptIndices_1 = \xi$
  for $i \leftarrow 2$ to $n_2$ do
      $C_i \leftarrow H_i^T w_{i+1}^2 (H_i^T w_i^2)^{-1}$      ▷ Projection of next location to see the measurement
      $Res_{i+1} \leftarrow w_{i+1}^2 - w_i^2 C_i$      ▷ Find error (Residual)
      $[\rho, \xi_i] \leftarrow max(Res_{i+1})$      ▷ Check for dominant ANN basis coefficient
      $H_{i+1} \leftarrow [H_i \xi_i]$      ▷ Update the $H$ matrix
      $OptIndices_i \leftarrow \xi_i$      ▷ Append the index location
  end for
```

Once the ANN-based DEIM algorithm ends, $OptIndices$ provides the complete list of optimal sensor locations with which the predictions of the Gappy ANN are obtained again, similar to the case of Gappy POD for POD-based DEIM. The results are investigated for any improvements in the predictions, as depicted in Figure 2.4.

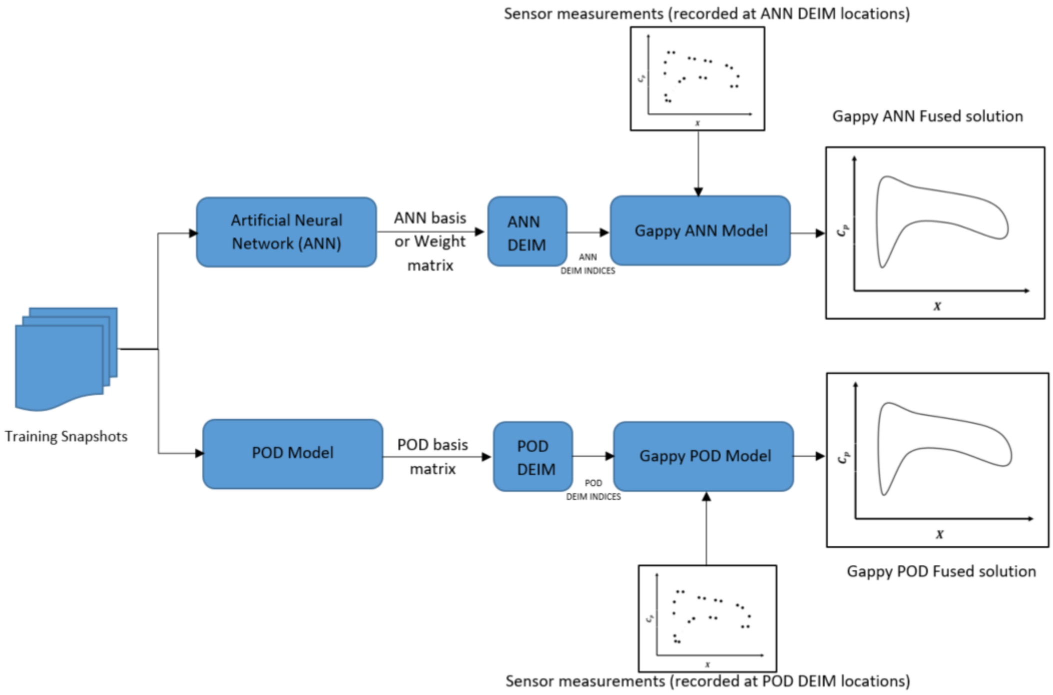
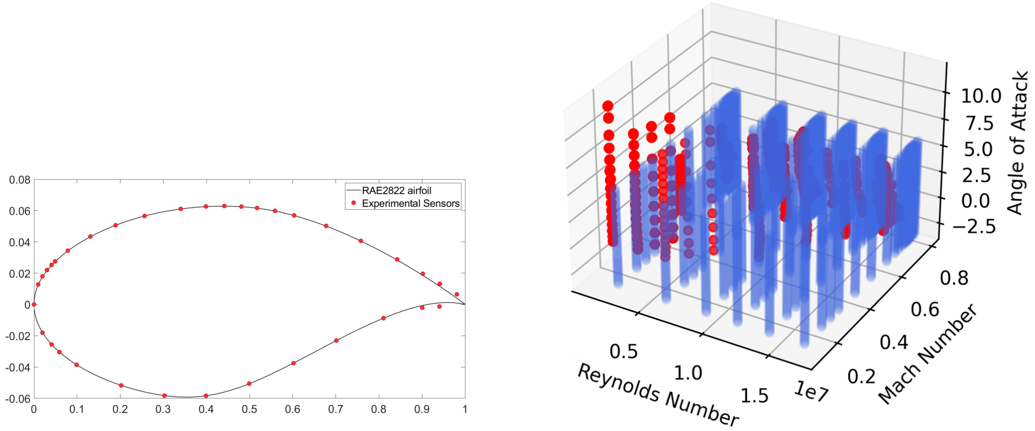


Figure 2.4: Workflow of data fusion approaches based on DEIM algorithm

3 Results and Observations

As part of testing the Gappy POD and Gappy ANN approaches, CFD simulations and wind tunnel test data are collected for the RAE2822 airfoil. The airfoil geometry is shown in Figure 3.1a. The RAE2822 is a transonic airfoil equipped with a thicker trailing edge. The airfoil was tested to cross-check results with legacy data and extend the range of the operating conditions with different combinations of Reynolds numbers (Re), Mach numbers (M), and angle of attack (α).



(a) The RAE2822 airfoil

(b) Parameter combination for which CFD data (blue) and wind tunnel data (red) are available

Figure 3.1: Test case and parameter combination for the sample generation in CFD and Wind tunnel testing

Pressure coefficients in Wind tunnel testing were recorded with 36 pressure taps located at the surface of the airfoil, as shown in Figure 3.1a. A total of 11466 RANS-CFD simulation results are gathered for similar flow conditions on a computational grid of 531 points using the DLR flow solver TAU with the SST turbulence model. The TAU Code is widely used in the European aerospace sector, and validations of the code are available in the literature [39]. The parameter combination in Figure 3.1b shows the different flow conditions used to generate the Wind tunnel testing and CFD simulation results.

3.1 Issues in Training

Unlike POD, SNN needs input and target data pairs to train the model. This thesis also deals with wind tunnel experiment data for which the complete surface solution is unavailable. As this target information unavailable, the following training configurations can be considered:

1. **Wind tunnel sensor and CFD data:** The idea is that the full surface solution for the wind tunnel sensor data can be generated for the same flow condition in which

the sensor measurements are generated using the CFD simulation. The available sensor measurement can be used as the input and the CFD solution as the target while training the network. Following this configuration might result in a drawback as one may have to consider the generated CFD data as an absolute truth, which is undesirable in this case.

2. **CFD data only:** The input data is created from the existing CFD solution based on the information at the nearest neighbor of the sensor locations in the CFD mesh. The network will be trained using these input data against the full CFD solution as the target. As in the first configuration, this setup also has a drawback. The SNN model will only be trained on the CFD data, and existing sensor measurement data will never be used. However, the objective of this thesis is to investigate the different data fusion methods; this configuration is preferable as it is also closely related to the Gappy POD setup.

Based on the above-decided configuration, the sampling management is done as follows. Around 70% of the CFD, data is used for training, 20% for validation, and 10% for testing. The SNN is built using PyTorch, whereas the gappy POD model has been imported from the fusion capability of the SMARTy [2], which is the DLR's Python-based Surrogate Modeling for AeRodata Toolbox. The Gappy POD model is created with a POD subspace of 40 modes with a relative information content (RIC) of 99.9947% using the training and validation dataset. Based on the number of modes selected while creating a Gappy POD basis, the same number of neurons l_2 =40 are chosen to compare the results effectively. Other parameters, namely the learning rate and activation function, are chosen based on the study [5], which determined these hyperparameters using the hyperparameter optimization framework Optuna. The list of all other essential parameters can be seen in Table 3.1. The SNN model is trained for 7800 epochs with a batch size of 64 samples. The MSE loss function has been used throughout for validation and training purposes.

Parameter	Value
Size of the input Layer s	36
Size of the first hidden layer l_1	110
Size of the second hidden layer l_2	40
Size of the output layer N	531
Learning rate	0.00248
Activation function	ReLU
Loss function	MSE
Optimization function	Adamax

Table 3.1: Hyperparameters of the SNN and their corresponding values

3.2 Results of the Gappy methods

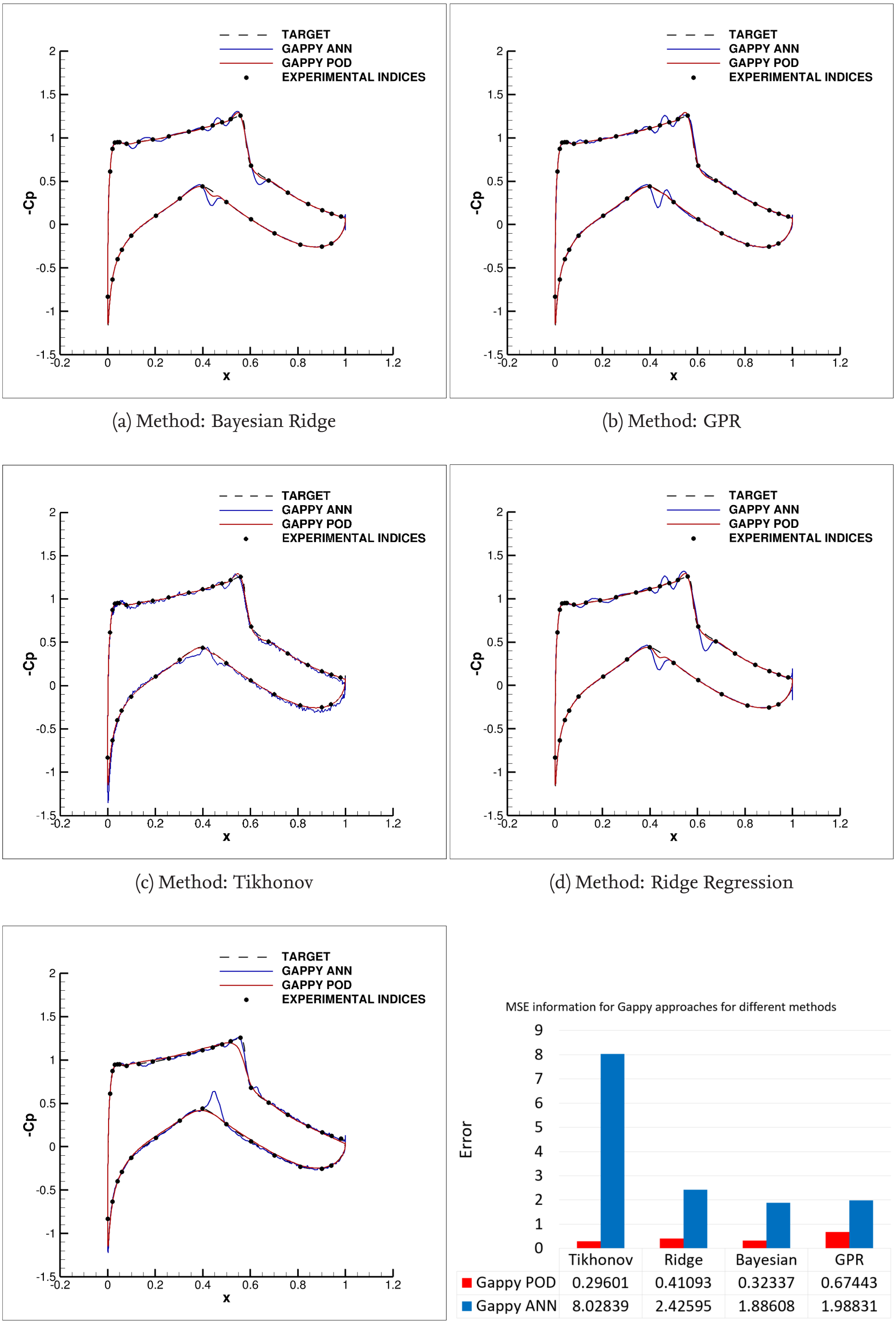
3.2.1 Prediction of CFD data

Once the training step is finished, the weight matrix is obtained from the final hidden layer of the SNN model. This weight matrix is later used in the Gappy ANN approach.

To understand which methods in the literature are best suited to solve the least-square problem, the result of Gappy ANN and Gappy POD for a flow condition of $Re = 12000000.0$, $M = 0.78$ and $\alpha = 3.3^\circ$ is analysed.

The results of the analysis are shown in Figures 3.2. The target information for this flow condition is obtained from the CFD solutions and shown in a black dashed line. The Gappy ANN results are in blue, and the Gappy POD is in red. The experimental sensor locations are given as black dots. Figure 3.2a shows the result of the Bayesian ridge method. The Gappy POD works very well with this method as it fits nicely with the target information by capturing the pressure peak, shock wave location, and strength compared to the Gappy ANN result, which suffers from fluctuations at the airfoil's upper surface. Figure 3.2b shows the results obtained for the Gaussian process regression (GPR) method. Like the first case, the Gappy POD accurately predicts the pressure peak, location, and strength of the shock. In the case of the Gappy ANN approach, the fluctuations continue to occur at the airfoil's upper and lower surface in its prediction. A slight wiggling effect can also be seen in the solution of the Gappy ANN near the trailing edge, which causes a little discrepancy in its prediction. The next method tested is Tikhonov, with a regularization of 0.05, and the results can be seen in Figure 3.2c. The Gappy POD solution fit smoothly to the target with a slight overshoot of pressure peak at the beginning of the shock wave compared to the solution of the Gappy ANN, which suffers from the high unphysical wiggling effect. Due to this, a significant data difference exists between the prediction and target, proving that this method does not suit well. Figure 3.2d displays the results for the ridge regression method. The prediction result of the Gappy ANN is almost similar to that of Bayesian ridge regression but with more difference at the shock wave's start and end locations. Although the Gappy POD accurately captures the shock wave's strength, it overshoots the pressure peak before the shock. The last method investigated is SVR; the result can be seen in Figure 3.2e. The Gappy POD accurately predicts the pressure peak but fails to capture the location and strength of the shock wave. Though Gappy ANN can accurately predict the pressure peak, location, and strength of the shock wave, the solution suffers from an unphysical wiggling effect at the upper and lower surface of the airfoil. These observations illustrate that SVR is unsuitable for both approaches. The MSE information is also taken for the complete test sample set for all the methods to compare the results quantitatively. From the MSE analysis, as shown in Figure 3.2f, the Bayesian ridge regression, Ridge regression, and GPR seem to work better for both approaches, out of which the Bayesian ridge regression fuses the solution for both the approaches with minimal error. Based on this finding, the results of further analysis are recorded using the Bayesian ridge method.

Once the method of solving the Gappy problem has been determined, both the Gappy approaches are analyzed for different flow conditions. The results can be seen in Figure 3.3. The first condition investigated is with $Re = 2700000.0$ and $M = 0.1$ and $\alpha = -0.2^\circ$ as shown in Figure 3.3a. The Gappy POD can give the best prediction accuracy with a smooth fit to the target. In contrast, although Gappy ANN captures the right physical trend, the solution suffers from fluctuations at the upper and lower surface of the airfoil. In the following case, a flow condition of $Re = 6500000.0$ and $M = 0.72$ and $\alpha = -2.2^\circ$ is investigated, and the result can be seen in Figure 3.3b. Both the Gappy ANN and Gappy POD fail to capture the correct pressure peak near the wake of the shock wave. In addition, both approaches fail to capture the shock wave's location and strength accurately. Like in



(a) Method: Bayesian Ridge

(b) Method: GPR

(c) Method: Tikhonov

(d) Method: Ridge Regression

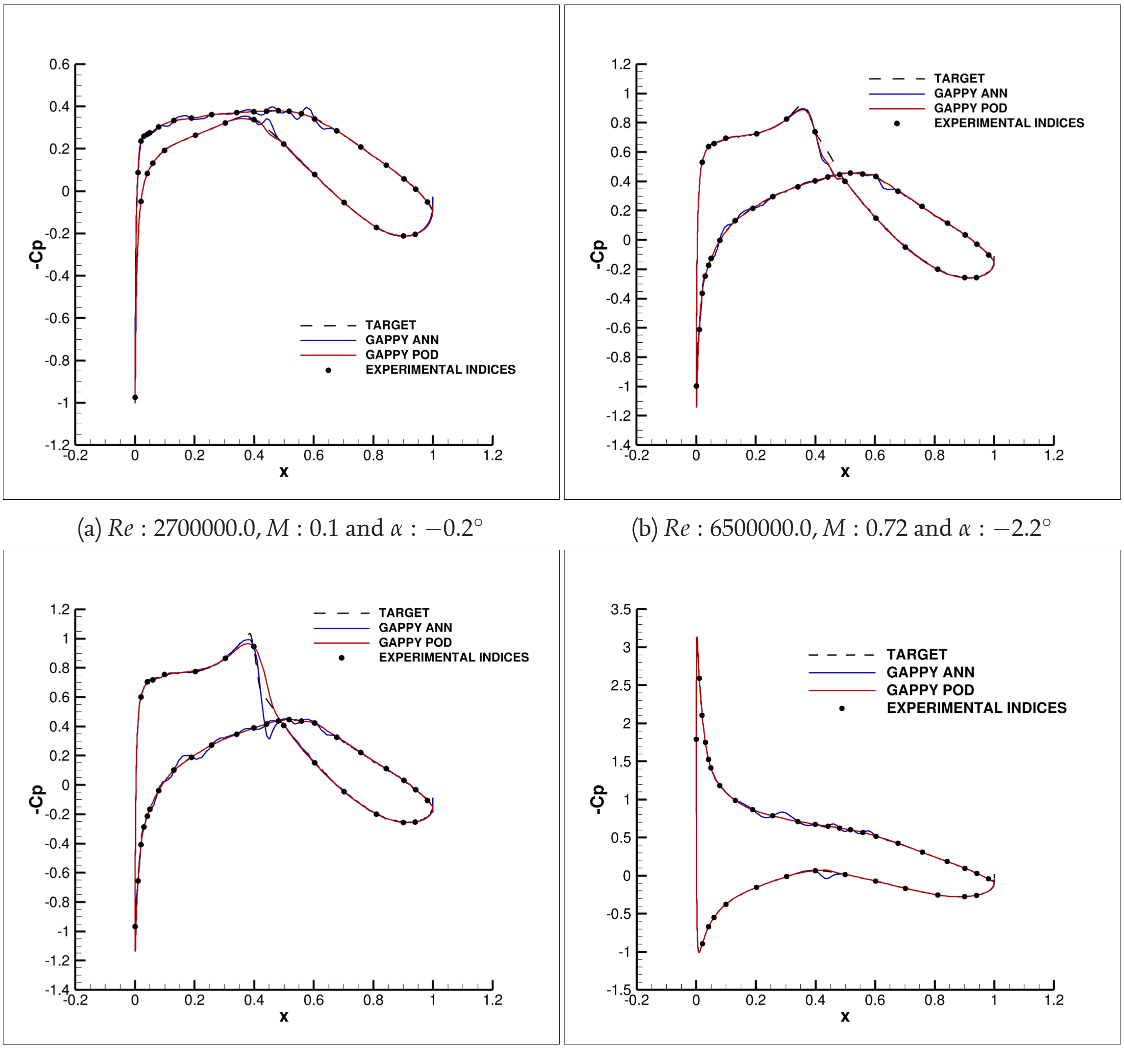
(e) Method: SVR

	Tikhonov	Ridge	Bayesian	GPR
🟥 Gappy POD	0.29601	0.41093	0.32337	0.67443
🟦 Gappy ANN	8.02839	2.42595	1.88608	1.98831

(f) MSE for all the test samples for all the methods

Figure 3.2: Results of Gappy ANN and Gappy POD for different methods at flow condition $Re = 12000000.0$, $M = 0.78$ and $\alpha = 3.3^\circ$

the first case, the unphysical fluctuations can be seen in the solution of the Gappy ANN. The next investigated case is of $Re = 6500000.0$ and $M = 0.725$ and $\alpha = -2.4^\circ$. As shown in Figure 3.3c, the fused solution of the Gappy POD could not accurately predict the shock wave's pressure peak, location, and strength. In contrast, the prediction of Gappy ANN almost gets the location of the shock wave but overshoots its ending. The fused solution of Gappy ANN suffers from wiggling at the lower surface of the airfoil, which is undesirable. Figure 3.3d shows the result of the last case with a flow condition of $Re = 12000000.0$ and $M = 0.2$ and $\alpha = 5.0^\circ$. Both approaches can accurately predict the suction pressure peak, especially Gappy POD, which provides accurate results with an excellent fit. In contrast, the Gappy ANN gives the real physical trend, but accuracy is impacted due to the wiggling effect at the airfoil's upper surface. Overall, the solution obtained through the Gappy POD approach fits more accurately to the CFD target data from all the investigated cases than Gappy ANN predictions.



(a) $Re : 2700000.0, M : 0.1$ and $\alpha : -0.2^\circ$

(b) $Re : 6500000.0, M : 0.72$ and $\alpha : -2.2^\circ$

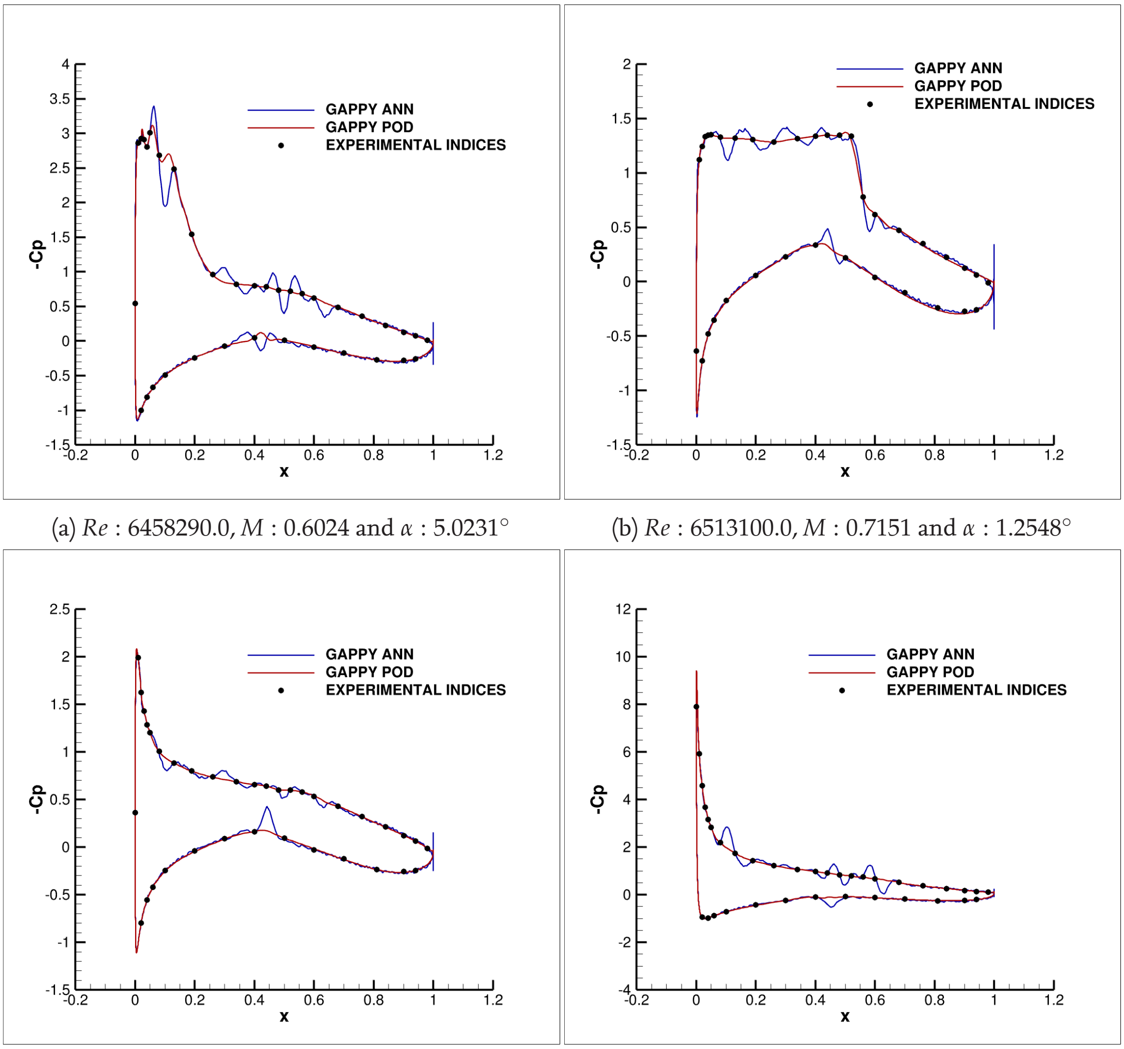
(c) $Re : 6500000.0, M : 0.725$ and $\alpha : -2.4^\circ$

(d) $Re : 12000000.0, M : 0.2$ and $\alpha : 5.0^\circ$

Figure 3.3: Results of Gappy ANN and Gappy POD for different flow conditions for CFD target data

3.2.2 Prediction of Wind Tunnel data

The investigation has extended to test these Gappy approaches based on the wind tunnel experimental data (WTT). Considering the issues associated with the training configuration (section 3.1), WTT data sets were neither part of the SNN training data sets nor used while creating the POD model. The WTT and CFD data will have different mechanisms and strategies to follow. For example, the solution in CFD simulation is captured over high-resolution computational grid points, whereas WTT uses a few discrete points. Hence testing the Gappy approaches on the WTT data will help to understand their sensitivities towards wind tunnel effects. Therefore four test samples are selected from the WTT ranging from high to low *Re*, and the results can be seen in Figure 3.4.



(a) Re : 6458290.0, M : 0.6024 and α : 5.0231°

(b) Re : 6513100.0, M : 0.7151 and α : 1.2548°

(c) Re : 9699360.0, M : 0.3945 and α : 1.9833°

(d) Re : 2866200.0, M : 0.1941 and α : 7.7892°

Figure 3.4: Results of Gappy ANN and Gappy POD for different flow conditions - Wind Tunnel experimental data

The target information is unavailable for WTT samples for a qualitative comparison of the result, as mentioned previously. However, the effectiveness of these approaches for the WTT samples is analyzed based on the result's trend corresponding to experimental sensor locations. In Fig 3.4a, the result obtained for Gappy ANN and Gappy POD for a flow condition of $Re = 6458290.0$ and $M = 0.6024$ and $\alpha = 5.0231°$ is given. The Gappy POD fits the sensor location more accurately than the Gappy ANN. The trend of the Gappy POD solution is smooth compared to the wiggling and fluctuation in the Gappy ANN result. In the following Figure 3.4b, the result for the flow condition $Re = 6513100$and $M = 0.7151$ and $\alpha = 1.2548°$ is shown. The Gappy POD again works better than Gappy ANN for this case, with a smooth fit across all the experimental sensor locations. In contrast, the solution of the Gappy ANN suffers from strong unphysical fluctuations while predicting the pressure plateau near the shock location. A slight wiggling effect can also be seen between some sensors, especially near the trailing edge. The subsequent Figure 3.4c shows the result for a flow condition $Re = 9699360.0$ and $M = 0.3945$ and $\alpha = 1.9833°$. A smooth trend with a better fit is obtained from the Gappy POD solution, which accurately matches the WTT data. The Gappy ANN gives a slightly improved prediction with a small wiggling effect between the sensors at both the upper and lower surface of the airfoil. The last case investigated is a flow condition $Re = 2866200.0$ and $M = 0.19414$ and $\alpha = 7.7892°$ as displayed in Figure 3.4d. The Gappy POD performs better by accurately matching the WTT data. Although the Gappy ANN solution looks slightly improved with less wiggling effect, strong fluctuations continue to occur at the airfoil's upper surface, which hampers its accuracy. From the above observations, it is noticeable that Gappy POD works better for the WTT data set with an accurate fit. In all the above-investigated cases, it accurately predicted the location and strength of the shock and pressure peaks with a smooth trend. Gappy ANN is not effectively capturing the details like shock locations and pressure peaks due to the wiggling effect and strong fluctuations.

3.3 Results of DEIM-based optimal sensor placement strategy

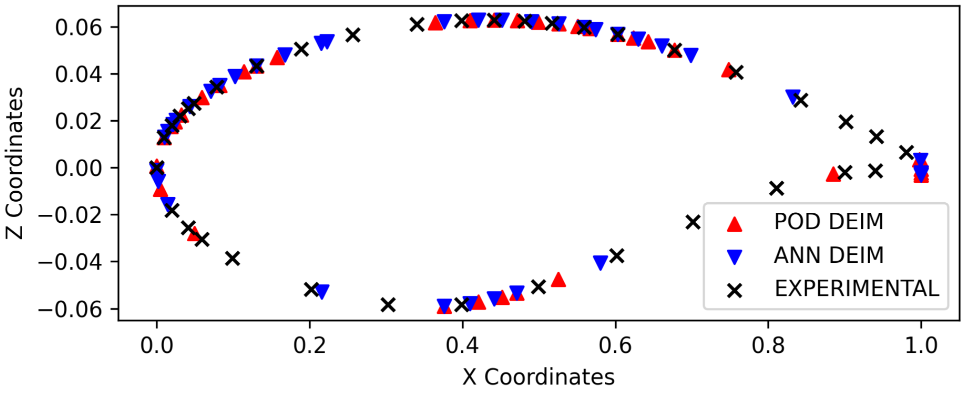
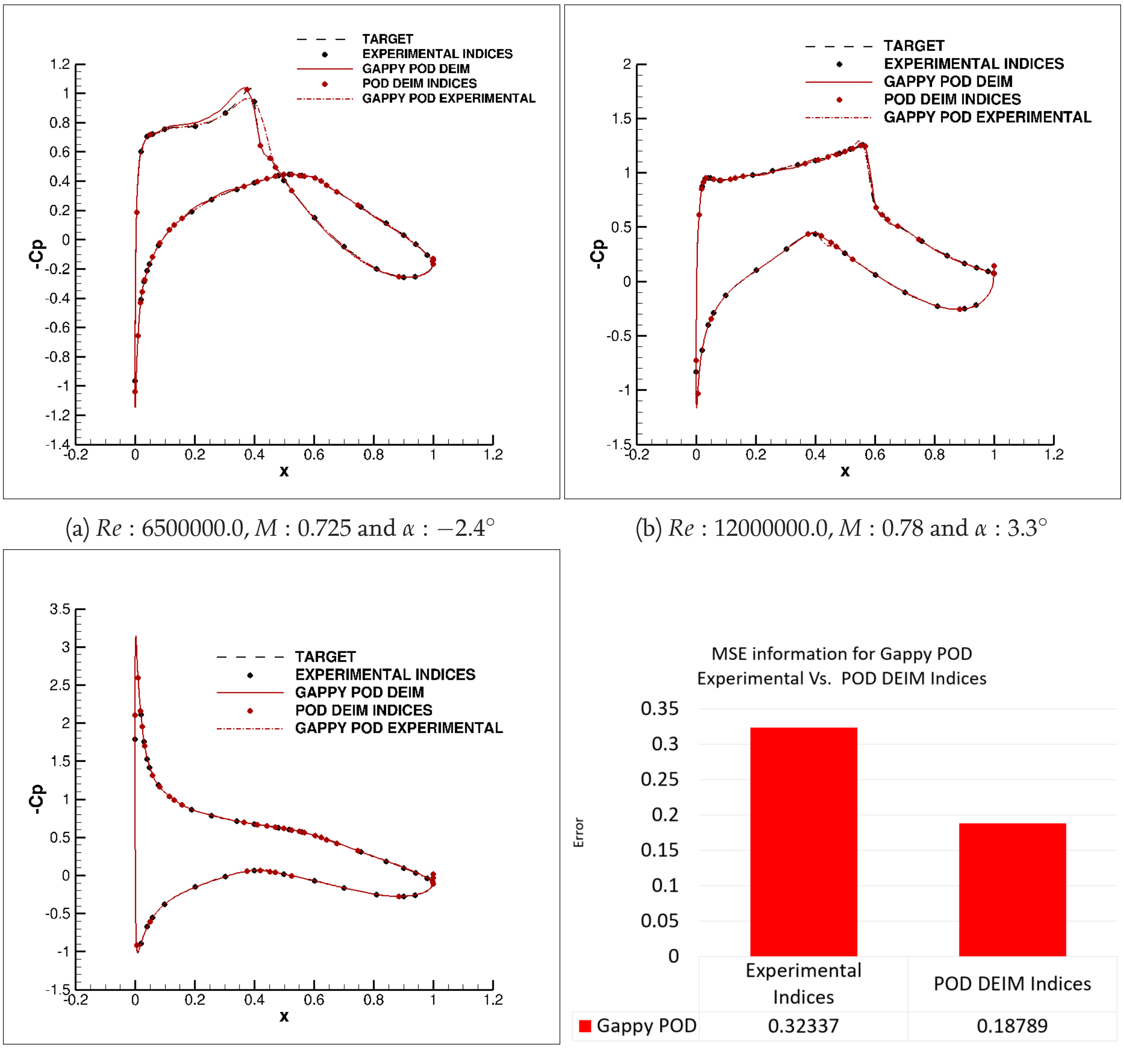


Figure 3.5: Experimental Vs. DEIM indices

The investigation of the data fusion techniques is extended to find the optimal sensor placement strategy based on the POD-reduced space and the space generated by the ANN using the DEIM algorithm. At first, an already created POD model with 40 modes is considered for this strategy. The DEIM algorithm iterates over all the POD basis vectors to find the indices of the dominant POD modes to give the optimal sensor locations. Only the first 36 indices are considered from the list of DEIM interpolation points to compute the Gappy POD solution. Afterward, the POD-reduced space is replaced with the existing ANN weight matrix. Once again, the DEIM algorithm is applied to the ANN solution space to get the optimal sensor locations. The calculated indices are used in the Gappy ANN approach to check for improvements in its prediction. The calculated POD DEIM and ANN DEIM indices are illustrated in Figure 3.5, and their respective solutions are discussed in detail below.

3.3.1 Results of Gappy POD based on POD-based DEIM approach



(a) Re : 6500000.0, M : 0.725 and α : $-2.4°$

(b) Re : 12000000.0, M : 0.78 and α : $3.3°$

(c) Re : 12000000.0, M : 0.2 and α : $5.0°$

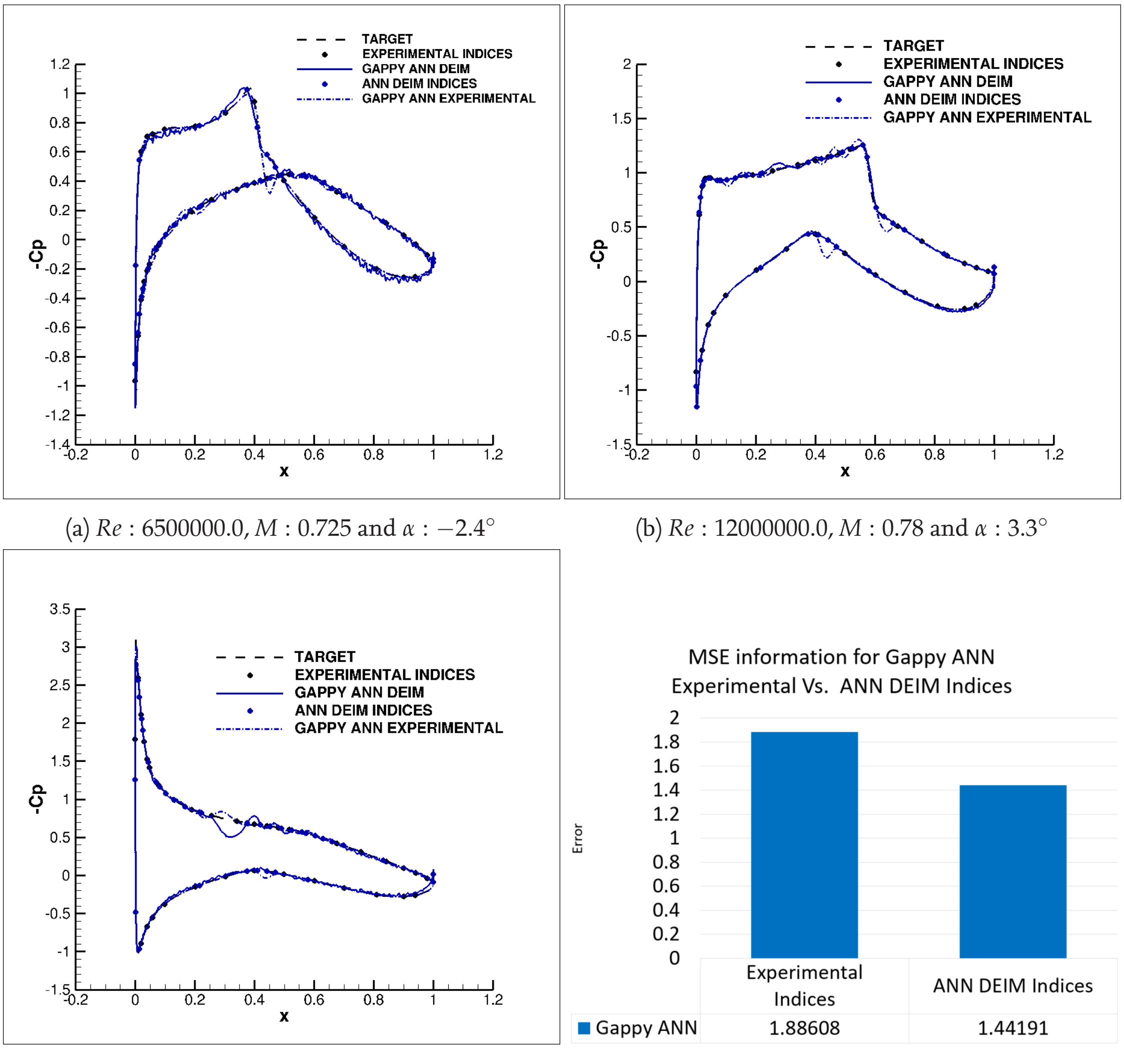
(d) MSE information of Gappy POD for experimental Vs. POD DEIM indices

Figure 3.6: Results of Gappy POD based on the POD DEIM indices for CFD target data

The effectiveness of the POD DEIM indices on the Gappy POD approach is analyzed by predicting the solution for three cases, and the results are illustrated in Figure 3.6. The target information is shown in black dashed lines, and the experimental sensor information is in black circles. The indices obtained from POD DEIM are displayed in red circles, and the corresponding solution is given as a red line. For comparison, the result of Gappy POD for experimental indices is given in red dash-dot line. The first analyzed case in Figure 3.6a is of flow condition *Re*: 6500000.0, *M*: 0.725, and α: -2.4°. With the POD DEIM indices, the Gappy POD can accurately predict the pressure peaks, strength, and location of the shock wave, which were improper in the prediction using the experimental indices. Despite this improvement in the solution, the Gappy POD cannot accurately capture the pressure plateau near the shock's beginning, which was recorded accurately with the experimental indices. The next case considered is of flow condition *Re*: 12000000.0, *M*: 0.78 and α: 3.3° as displayed in Figure 3.6b. The Gappy POD has given an excellent fit using the POD DEIM indices removing all the slight discrepancies near the shock location, which were present in the solution for experimental indices. The last case investigated is *Re*: 12000000.0, *M*: 0.2 and α: 5.0° as depicted in Figure 3.6c. The Gappy POD retains accuracy in its result from experimental and POD DEIM indices without any discrepancy between both solutions. The MSE information is also calculated for the complete test samples for quantitative comparison, and the result can be seen in Figure 3.6d. The MSE information shows that the performance of the Gappy POD approach increased with a reduced error for the POD DEIM indices.

3.3.2 Results of Gappy ANN based on ANN-based DEIM approach

The results of the Gappy ANN approach based on ANN DEIM indices are also analyzed for any possible improvements, as done for the Gappy POD approach. In Figure 3.7, the target information is given in black dashed lines, and the experimental sensor information is in black circles. The POD DEIM is displayed in blue circles, and the corresponding solution obtained is illustrated in the blue line. For comparison, the result of Gappy ANN for experimental indices is given in blue dash-dot line. The case *Re*: 6500000.0, *M*: 0.725, and α: -2.4° is analyzed first. As shown in Figure 3.7a, the Gappy ANN prediction suffers from a strong wiggling effect. Along with this effect, discrepancies exist at the airfoil's upper surface before the shock and near the trailing edge. The solution of ANN DEIM shows no significant improvement compared to the one obtained using the experimental indices for this case. The case *Re*: 12000000.0, *M*: 0.78, and α: 3.3° is analyzed next. As shown in Figure 3.7b, the Gappy ANN prediction for the ANN DEIM indices gives an improved fit with a better prediction of the pressure peaks, strength, and location of the shock wave compared to the fused solution obtained using the experimental indices. A slight discrepancy exists at the airfoil's upper surface and near the trailing edge due to the absence of ANN DEIM indices in those areas. The last case investigated is *Re*: 12000000.0, *M*: 0.2 and α: 5.0° and the result can be seen in Figure 3.7c. For this particular test case, a wiggling effect can be seen in all the areas of the airfoil. In addition, the solution of the Gappy ANN fails to predict the suction pressure peak, which was captured accurately using the experimental indices. Nevertheless, to understand the performance of the Gappy ANN approach for ANN DEIM, MSE information is computed for complete test samples. The MSE information, as shown in Figure 3.7d, indicates that the Gappy ANN has shown significantly improved performance compared to the experimental indices.



(a) Re : 6500000.0, M : 0.725 and α : $-2.4°$

(b) Re : 12000000.0, M : 0.78 and α : $3.3°$

(c) Re : 12000000.0, M : 0.2 and α : $5.0°$

(d) MSE information of Gappy ANN for experimental Vs. ANN DEIM indices

Figure 3.7: Results of Gappy ANN based on the ANN DEIM indices for CFD target data

3.3.3 DEIM algorithm analysis for various POD modes

The DEIM algorithm uses the POD subspace to find the location of the optimal indices. It is interesting to analyze the behavior of the DEIM algorithm based on different numbers of POD modes. For this purpose, three different POD mode cases are considered. A POD-reduced space is generated with 250 POD modes in the first case. The DEIM algorithm is applied to this reduced space to compute the optimal sensor

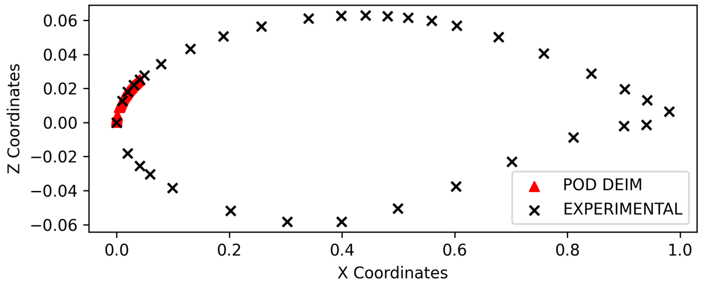
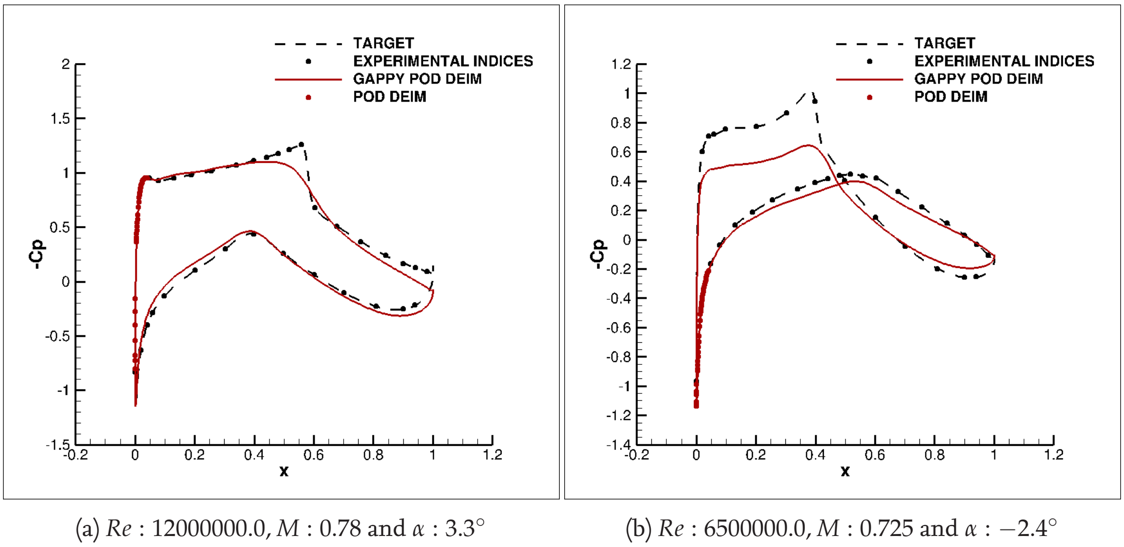


Figure 3.8: Experimental Vs. DEIM indices for 250 POD modes



(a) Re : 12000000.0, M : 0.78 and α : 3.3°

(b) Re : 6500000.0, M : 0.725 and α : −2.4°

Figure 3.9: Results of Gappy POD for POD DEIM indices (POD modes = 250)

locations. The first 36 locations are selected from the complete result list. As shown in Figure 3.8, the new locations recommended by the DEIM algorithm concentrate mainly on the leading edge without covering the entire airfoil area. In the next step, these new locations are used in the Gappy POD approach to predict the flow solution, and the results are displayed in Figure 3.9. The computed Gappy POD solutions lose their accuracy at all regions except the leading edge. The reason may be due to singularity in the POD basis matrix, which weekly contributes while finding the dominant modes.

In the subsequent case POD model is created with only 100 modes. After applying the DEIM algorithm, as illustrated in Figure 3.10, the newly computed POD DEIM indices are plotted against the experimental indices to check their effectiveness in covering key areas. As seen in case one, most of the new recommended locations concentrate at the leading edge, with few assigned at the trailing edge missing all other areas. Hence, the respective Gappy POD solutions for these indices do not significantly improve their prediction, as shown in Figure 3.11.

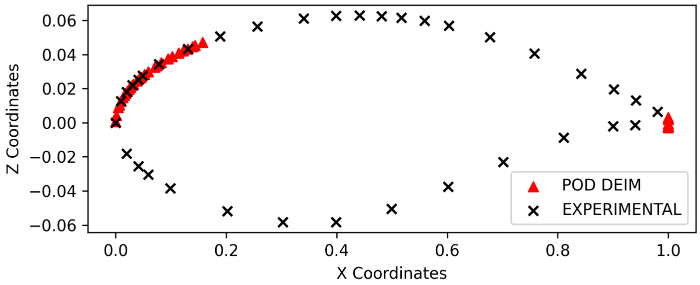
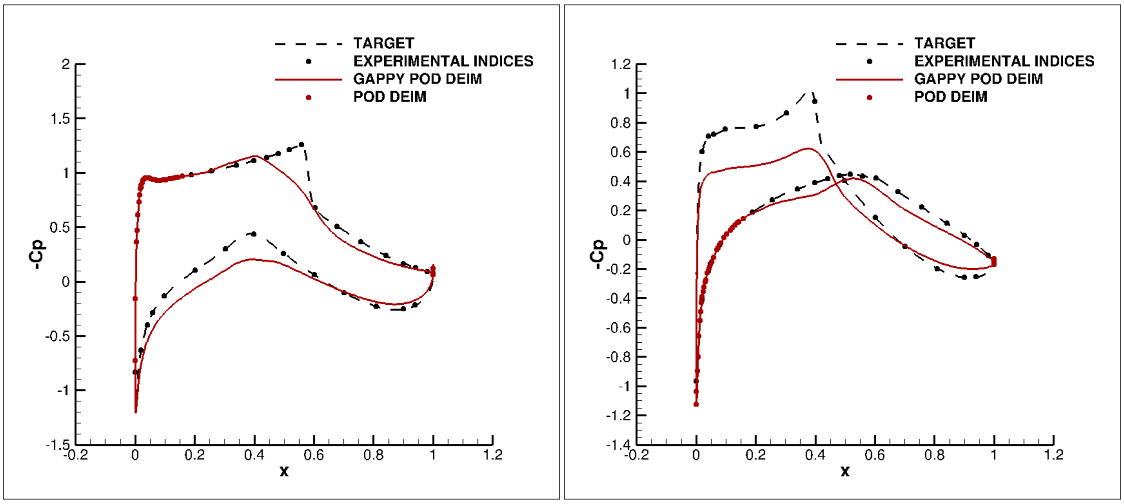


Figure 3.10: Experimental Vs. DEIM indices for 100 POD modes



(a) Re : 12000000.0, M : 0.78 and α : 3.3°

(b) Re : 6500000.0, M : 0.725 and α : −2.4°

Figure 3.11: Results of Gappy POD for POD DEIM indices (POD modes = 100)

As a solution, the POD model is created with only 50 modes to refine the POD subspace in the last case. Figure 3.12 shows that the new recommended locations obtained from the DEIM offer adequate improvements as they spread across the airfoil, covering the critical areas. The Gappy POD solutions obtained for these new recommendations also confirm the improvements in their prediction with better accuracy, as displayed in Figure 3.13.

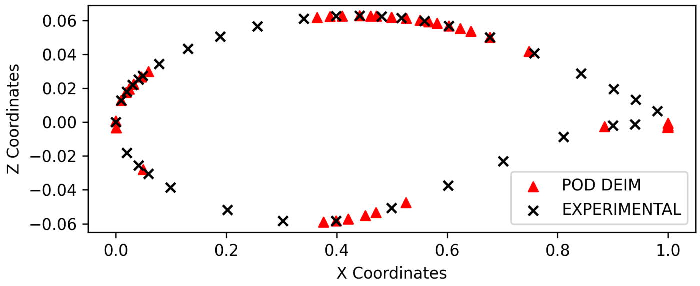
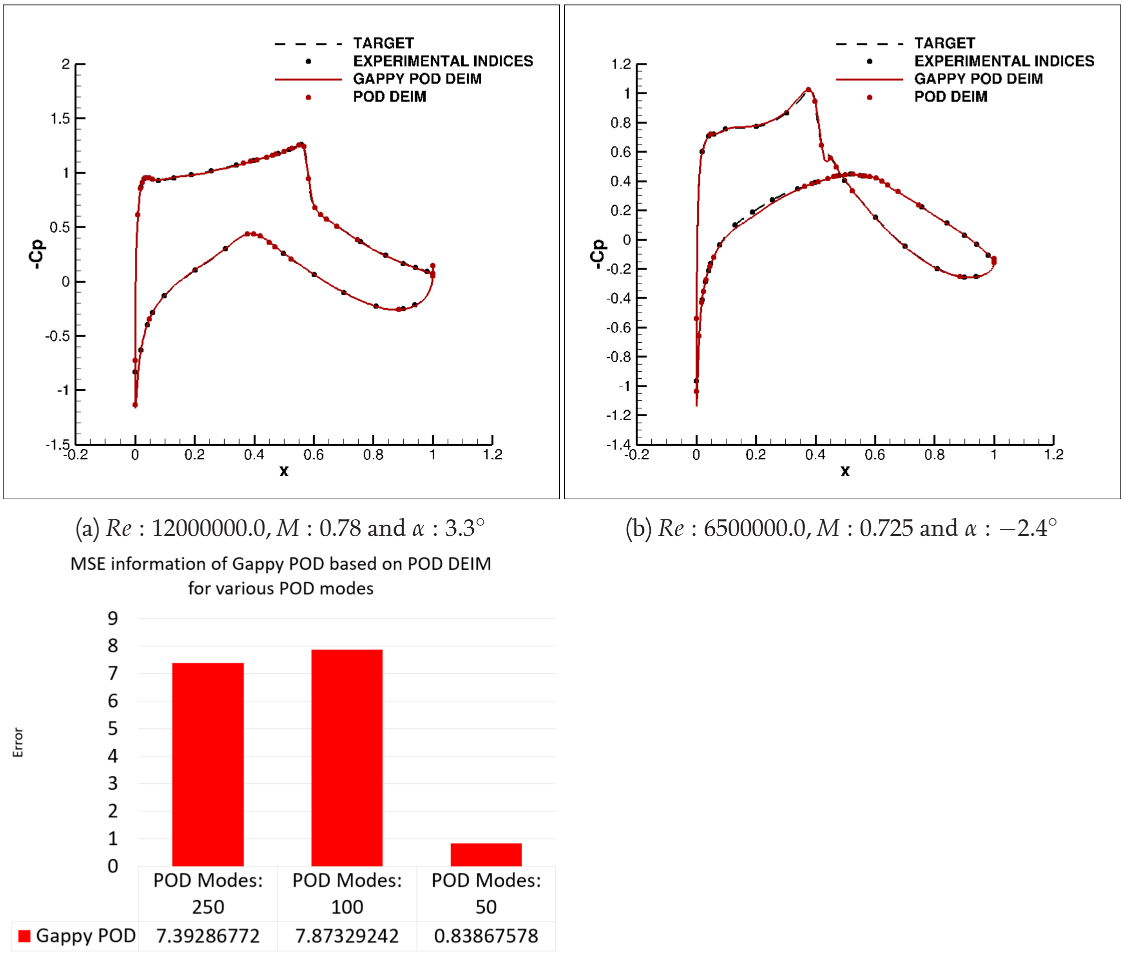


Figure 3.12: Experimental Vs. DEIM indices for 50 POD modes

The MSE is also calculated for the complete test set for all the above cases. Figure 3.13c displays that the MSE loss reduces concerning the fewer modes chosen when creating the POD model. Hence this analysis demonstrates the effectiveness of the DEIM algorithm for various POD modes cases and the performance of Gappy POD for these cases.



(a) Re : 12000000.0, M : 0.78 and α : 3.3°

(b) Re : 6500000.0, M : 0.725 and α : −2.4°

(c) MSE information of Gappy POD based on POD DEIM for various POD modes

Figure 3.13: Results of Gappy POD for POD DEIM indices (POD modes = 50) and MSE information for different modes

4 Conclusion and Outlook

4.1 Conclusion

This thesis investigated the POD-based and a newly proposed ANN-based Gappy approach. The proposed Gappy ANN alters the Gappy POD by replacing the POD solution space with the ANN space defined through the weight matrix from the final hidden layer. This effectively alters the solution within which the flow solutions are fused. The performance of Gappy POD and Gappy ANN was tested on a test case with high-quality data comprising wind tunnel experimental data and the CFD simulation data. The DEIM algorithm is employed on the POD subspace and ANN solution space to find the optimal sensor location by computing the indices of the dominant modes. These algorithms are referred to as POD DEIM and ANN DEIM. Based on the requirement, the first 36 locations are selected from the DEIM interpolation points and used in the respective Gappy approaches instead of the experimental indices to investigate potential improvements in the fused solutions.

From the analysis, the Gappy POD approach provided better results than the proposed Gappy ANN approach. Gappy POD fused the solution more accurately for the CFD and WTT measurement datasets with an excellent fit and a smooth trend. In contrast, the accuracy of the Gappy ANN solutions is degraded due to unphysical fluctuations and wiggling effects. These effects were strong, especially for the WTT dataset, reducing their accuracy even further. The DEIM algorithm is applied on top of the POD and ANN solution space to find the optimal locations, potentially yielding more accurate flow reconstructions. The new indices from DEIM are used in their respective Gappy approaches to investigate potential improvements. The result of the Gappy POD shows some discrepancy for new indices compared to the experimental. Nevertheless, the MSE information calculated for the entire test set confirms improvements in the Gappy POD prediction. The Gappy ANN for ANN DEIM also improved its prediction, but the wiggling effects continued to appear strongly in the fused solutions. Overall, in both CFD and WTT data prediction, the Gappy POD approach has performed better than the Gappy ANN.

4.2 Outlook

Although an extensive analysis for selecting the correct hyperparameters is done while implementing the Gappy POD and Gappy ANN models, depending on the test data, one may have to address this problem in every analysis. Especially for the Gappy ANN model, the number of neurons in the hidden layers and other training parameters like epochs, learning rate, and the optimizer play a substantial role. The accuracy of the solution varies for every combination of the said hyperparameters; standardizing the selection of these can be of interest in future studies. Obtaining the weight matrix from the final hidden layer is of utmost interest to increase the prediction accuracy of the Gappy ANN by removing the fluctuation and wiggling effect. One can incorporate methods like Adaptive weight initialization or Xavier weight initialization to initialize weights. Even though first-time weights are assigned randomly, sometimes ANN may converge in local minima, leading to errors. Hence, supplying optimal initial weights may reduce this risk. The other possible measures are applying a dropout ratio, regularizing weight penalties, and employing early stopping, which may enhance the training process.

Choosing the correct technique to solve the Gappy least-square problem may improve the solution accuracy. As discussed, one can treat this Gappy problem as a regression problem, and many techniques are available in the literature to solve this regression problem. Selecting the proper hyperparameters like kernel function and regularization concerning these regression techniques certainly influences the accuracy of the solution. There is still potential for combining various other regression techniques with the Gappy ANN to test its performance in future studies.

Moreover, further work needs to be done regarding integrating known uncertainties like measurement uncertainties due to sensor inaccuracy in the wind tunnel and errors in the computational data due to modeling or convergence. These uncertainties can be quantified by, e.g., data analysis or expert knowledge.

Bibliography

[1] Rimah Amami, Dorra Ben Ayed, and Noureddine Ellouze. “Practical Selection of SVM Supervised Parameters with Different Feature Representations for Vowel Recognition”. In: CoRR abs/1507.06020 (July 2015). URL: http://arxiv.org/abs/1507.06020.

[2] Philipp Bekemeyer et al. “Data-Driven Aerodynamic Modeling Using the DLR SMARTy Toolbox”. AIAA Aviation 2022 Forum. Ed. by AIAA. American Institute of Aeronautics and Astronautics, Inc., June 2022, pp. 1–19. URL: https://elib.dlr.de/191797/.

[3] G Berkooz, PJ Holmes, and John Lumley. “The Proper Orthogonal Decomposition in the Analysis of Turbulent Flows”. In: Annual Review of Fluid Mechanics 25 (Nov. 2003), pp. 539–575. DOI: 10.1146/annurev.fl.25.010193.002543.

[4] Anna Bertram. “Data-driven variable-fidelity reduced order modeling for efficient vehicle shape optimization”. PhD thesis. Nov. 2018. DOI: 10.24355/dbbs.084-201811231243-0.

[5] Anna Bertram. “Outcomes of the DLR-Airbus Patenschaft “Development of Future Aerodynamic Data Modelling Methodology” 2018–2022”. Ed. by DLR. Bericht des Instituts für Aerodynamik und Strömungstechnik. Sept. 2022. URL: https://elib.dlr.de/188429/.

[6] Anna Bertram, Philipp Bekemeyer, and Matthias Held. “Fusing Distributed Aerodynamic Data Using Bayesian Gappy Proper Orthogonal Decomposition”. AIAA AVIATION 2021 FORUM. Aug. 2021. DOI: 10.2514/6.2021-2602. URL: https://arc.aiaa.org/doi/abs/10.2514/6.2021-2602.

[7] Anna Bertram, Carsten Othmer, and Ralf Zimmermann. “Towards Real-time Vehicle Aerodynamic Design via Multi-fidelity Data-driven Reduced Order Modeling”. In: AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference. Jan. 2018. DOI: 10.2514/6.2018-0916. URL: https://arc.aiaa.org/doi/abs/10.2514/6.2018-0916.

[8] Anna Bertram and Ralf Zimmermann. “Theoretical investigations of the new Cokriging method for variable-fidelity surrogate modeling: Well-posedness and maximum likelihood training”. In: Advances in Computational Mathematics 44 (Dec. 2018), pp. 1693–1716. DOI: 10.1007/s10444-017-9585-1.

[9] Jiri Blazek, ed. Computational Fluid Dynamics: Principles and Applications. Third Edition. Oxford: Butterworth-Heinemann, 2015. ISBN: 978-0-08-099995-1.

[10] Tan Bui-Thanh, Murali Damodaran, and Karen Willcox. “Aerodynamic Data Reconstruction and Inverse Design Using Proper Orthogonal Decomposition”. In: AIAA Journal 42 (Aug. 2004), pp. 1505–1516. DOI: 10.2514/1.2159.

[11] Chih-Chung Chang and Chih-Jen Lin. “LIBSVM: A Library for Support Vector Machines”. In: ACM Trans. Intell. Syst. Technol. 2.3 (May 2011). ISSN: 2157-6904. DOI: 10.1145/1961189.1961199. URL: https://doi.org/10.1145/1961189.1961199.

[12] Saifon Chaturantabut and Danny Sorensen. “Nonlinear Model Reduction via Discrete Empirical Interpolation”. In: SIAM J. Scientific Computing 32 (Jan. 2010), pp. 2737–2764. doi: 10.1137/090766498.

[13] Kelly Cohen et al. “Effective Sensor Placements for the Estimation of Proper Orthogonal Decomposition Mode Coefficients in von Karman Vortex Street”. In: Journal of Vibration and Control 10 (Dec. 2004), pp. 1857–1880. doi: 10.1177/1077546304046035.

[14] Zlatko Drmac and Serkan Gugercin. “A New Selection Operator for the Discrete Empirical Interpolation Method—Improved A Priori Error Bound and Extensions”. In: SIAM Journal on Scientific Computing 38.2 (2016), A631–A648. doi: 10.1137/15M1019271. url: https://doi.org/10.1137/15M1019271.

[15] Zlatko Drmač and Arvind Saibaba. “The Discrete Empirical Interpolation Method: Canonical Structure and Formulation in Weighted Inner Product Spaces”. In: SIAM Journal on Matrix Analysis and Applications 39.3 (2018), pp. 1152–1180. doi: 10.1137/17M1129635. url: https://doi.org/10.1137/17M1129635.

[16] N. Erichson et al. “Shallow neural networks for fluid flow reconstruction with limited sensors”. In: Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences 476 (June 2020), p. 20200097. doi: 10.1098/rspa.2020.0097.

[17] Richard Everson and Lawrence Sirovich. “Karhunen–Loève procedure for gappy data”. In: J. Opt. Soc. Am. A 12.8 (Aug. 1995), pp. 1657–1664. doi: 10.1364/JOSAA.12.001657.

[18] Alexander I. J. Forrester, Andras Sobester, and Andy J. Keane. Engineering Design via Surrogate Modelling - A Practical Guide. Wiley, 2008, pp. I–XVIII, 1–210. isbn: 978-0-470-06068-1.

[19] Thomas Franz and Matthias Held. “Data Fusion of CFD Solutions and Experimental Aerodynamic Data”. ODAS 2017. 2017. url: https://elib.dlr.de/114707/.

[20] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. “Deep Sparse Rectifier Neural Networks”. Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics. Ed. by Geoffrey Gordon, David Dunson, and Miroslav Dudík. Vol. 15. Proceedings of Machine Learning Research. Fort Lauderdale, FL, USA: PMLR, Nov. 2011, pp. 315–323. url: https://proceedings.mlr.press/v15/glorot11a.html.

[21] Zhong-Hua Han and Stefan Görtz. “Hierarchical Kriging Model for Variable-Fidelity Surrogate Modeling”. In: AIAA Journal 50 (Sept. 2012), pp. 1885–1896. doi: 10.2514/1.J051354.

[22] Zhong-Hua Han, Stefan Görtz, and Ralf Zimmermann. “Improving variable-fidelity surrogate modeling via gradient-enhanced kriging and a generalized hybrid bridge function”. In: Aerospace Science and Technology 25 (Mar. 2013), pp. 177–189. doi: 10.1016/j.ast.2012.01.006.

[23] Zhong-Hua Han, Ralf Zimmermann, and Stefan Görtz. “A New Cokriging Method for Variable-Fidelity Surrogate Modeling of Aerodynamic Data”. 48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition. Jan. 2010. url: https://arc.aiaa.org/doi/abs/10.2514/6.2010-1225.

[24] Zhong-Hua Han, Ralf Zimmermann, and Stefan Görtz. "Alternative Cokriging Model for Variable-Fidelity Surrogate Modeling". In: AIAA Journal 50 (May 2012), pp. 1205–1210. DOI: 10.2514/1.J051243.

[25] Juncai He, Lin Li, and Jinchao Xu. "ReLU deep neural networks from the hierarchical basis perspective". In: Computers and Mathematics with Applications 120 (Aug. 2022), pp. 105–114. DOI: 10.1016/j.camwa.2022.06.006.

[26] Balaji Jayaraman, S M Abdullah Al Mamun, and Lu. "Interplay of Sensor Quantity, Placement and System Dimension in POD-Based Sparse Reconstruction of Fluid Flows". In: Fluids 4 (June 2019), p. 109. DOI: 10.3390/fluids4020109.

[27] Andy Keane. "Wing Optimization Using Design of Experiment, Response Surface, and Data Fusion Methods". In: Journal of Aircraft 40 (July 2003), pp. 741–750. DOI: 10.2514/2.3153.

[28] Diederik P. Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization". In: 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings (2015). Ed. by Yoshua Bengio and Yann LeCun. URL: http://arxiv.org/abs/1412.6980.

[29] Yuichi Kuya et al. "Multifidelity Surrogate Modeling of Experimental and Computational Aerodynamic Data Sets". In: AIAA Journal 49 (Feb. 2011), pp. 289–298. DOI: 10.2514/1.J050384.

[30] Krithika Manohar et al. "Data-Driven Sparse Sensor Placement for Reconstruction". In: IEEE control systems 38 (May 2018), pp. 63–. DOI: 10.1109/MCS.2018.2810460.

[31] J. Mercer. "Functions of Positive and Negative Type, and their Connection with the Theory of Integral Equations". In: Philosophical Transactions of the Royal Society of London Series A 209 (Jan. 1909), pp. 415–446. DOI: 10.1098/rsta.1909.0016.

[32] Michael Mifsud et al. "Fusing wind-tunnel measurements and CFD data using constrained gappy proper orthogonal decomposition". In: Aerospace Science and Technology 86 (Mar. 2019), pp. 312–326. URL: https://elib.dlr.de/126313/.

[33] Kevin Murphy. Machine Learning: A Probabilistic Perspective. Vol. 58. MIT Press, Jan. 2012.

[34] René Pinnau. "Model Reduction via Proper Orthogonal Decomposition". In: vol. 13. Jan. 2008, pp. 95–109. ISBN: 978-3-540-78840-9. DOI: 10.1007/978-3-540-78841-6_5.

[35] Carl Edward Rasmussen and Christopher K. I. Williams. Gaussian processes for machine learning. Adaptive computation and machine learning. MIT Press, 2006, pp. I–XVIII, 1–248. ISBN: 026218253X.

[36] S. Ashwin Renganathan, Kohei Harada, and Dimitri N. Mavris. "Aerodynamic Data Fusion Toward the Digital Twin Paradigm". In: AIAA Journal 58.9 (2020), pp. 3902–3918. DOI: 10.2514/1.J059203. URL: https://doi.org/10.2514/1.J059203.

[37] Jerome Sacks et al. "Design and analysis of computer experiments. With comments and a rejoinder by the authors". In: Statistical Science 4 (Jan. 1989), pp. 433–435. DOI: 10.1214/ss/1177012413.

[38] Masanori Sawa, Masatake Hirao, and Sanpei Kageyama. “Kernel Functions”. In: July 2019, pp. 1–17. ISBN: 978-981-13-8074-7. DOI: 10.1007/978-981-13-8075-4_1.

[39] Dieter Schwamborn, Thomas Gerhold, and Ralf Heinrich. “The DLR TAU-CODE: Recent Applications in Research and Industry”. ECCOMAS CFD 2006 CONFERENCE. auf CD. 2006. URL: https://elib.dlr.de/22421/.

[40] J Seidel et al. “Feedback control of a circular cylinder wake”. Journal of Aerospace Engineering. Vol. 223. 4. 2009, pp. 379–392. DOI: 10.1243/09544100JAERO407.

[41] Michael Wall, Andreas Rechtsteiner, and Luis Rocha. “Singular Value Decomposition and Principal Component Analysis”. In: In A Practical Approach to Microarray Data Analysis 5 (Sept. 2002), pp. 91–109. DOI: 10.1007/0-306-47815-3_5.

[42] Karen Willcox. “Unsteady Flow Sensing and Estimation via the Gappy Proper Orthogonal Decomposition”. In: Computers and Fluids 35.2 (2006), pp. 208–226. ISSN: 0045-7930. DOI: https://doi.org/10.1016/j.compfluid.2004.11.006. URL: https://www.sciencedirect.com/science/article/pii/S0045793005000113.

[43] Brian Williams et al. “Sequential Design of Computer Experiments for Constrained Optimization”. In: Statistical Modelling and Regression Structures: Festschrift in Honour of Ludwig Fahrmeir. Jan. 2010, pp. 449–472. ISBN: 978-3-7908-2412-4. DOI: 10.1007/978-3-7908-2413-1_24.

[44] Battalgazi Yildirim, C. Chryssostomidis, and George Karniadakis. “Efficient sensor placement for ocean measurements using low-dimensional concepts”. In: Ocean Modelling 27.3 (2009), pp. 160–173. ISSN: 1463-5003. DOI: https://doi.org/10.1016/j.ocemod.2009.01.001.

List of Figures

List of Tables