The 8th European Congress on Computational Methods in Applied Sciences and Engineering
ECCOMAS Congress 2022
5 – 9 June 2022, Oslo, Norway

TREE-BASED COMPARATIVE PREDICTION OF STEADY TRANSONIC FLOWS OVER A WING

A. GORGUES*, R. CASTELLANOS+, J. BOWEN AND E. ANDRÉS-PÉREZ

Theoretical and Computational Aerodynamics Branch, Flight Physics Department,
Spanish National Institute for Aerospace Technology (INTA)
Ctra. Ajalvir, km. 4. 28850 Torrejón de Ardoz, Spain
*PhD Student at University of Alcala
+PhD Student at University Carlos III de Madrid

Key Words: *Machine learning, Data mining, Aerodynamic analysis, Computational fluid dynamics, Reduced order modelling.*

Abstract. Machine learning entails powerful information processing algorithms that are relevant for modelling, optimization, and control of fluids. Currently, machine-learning capabilities are advancing at an incredible rate, and fluid mechanics is beginning to tap into the full potential of these powerful methods. Many tasks in fluid mechanics, such as reduced-order modelling, shape optimization and uncertainty quantification, may be posed as optimization and regression tasks. Machine learning can dramatically improve optimization performance and reduce convergence time.

In this paper, the potential of tree-based machine learning techniques for the aerodynamic prediction of pressure coefficients of an AIRBUS XRF1 aircraft wing-body configuration has been assessed. For this purpose, a dataset including computational fluid dynamics (CFD) simulations has been employed to train the different models, with and without the use of proper orthogonal decomposition (POD) and having their hyperparameters values optimized to obtain the optimal subspace. A deep comparison of decision tree regressors and random forest algorithms has been performed, showing that the random forest regressor model performs better on all configurations.

1 INTRODUCTION

Nowadays, the use of Computational Fluid Dynamics (CFD) simulations is a common practice in aeronautical industries due their level of maturity. Such companies use intensively CFD to estimate aerodynamic data for a configuration of interest (e.g., an aircraft component). The obtained data from CFD runs are commonly then analysed by experts that take certain decisions for instance, regarding the optimal design of the configuration or its optimal performance.

However, due the recent evolution of data-driven techniques, companies are now starting to store the results of the CFD simulations, so it is possible to use all these data with multiple purposes. One of these purposes, is to use the data for training a machine learning model that is capable to reproduce the behaviour of the CFD solver in terms of prediction of some aerodynamic features in a fast and reasonable accurate manner, opposite to the intensively computations, that the CFD codes usually require to execute. Machine learning models, which rely on being data-driven, arises long ago as a solution to reduce computational time, even on highly constrained configurations [1].

Coefficient prediction in aerodynamics is one of the main challenges to solve in fluid mechanics among others such as flow field prediction [2] or turbulence modelling [3].

In this paper, a comparison of different tree-based models for aerodynamic coefficients prediction is proposed. The selected test case is the XRF1 model, provided by AIRBUS. XRF1 is an Airbus-provided industrial standard multi-disciplinary research test/case representing a typical configuration for a long-range wide body aircraft. The XRF1 research test/case is used by Airbus to engage with external partners on development and demonstration of relevant capabilities / technologies.

This paper summarizes part of the work done within the GARTEUR project ML4AERO [4] where INTA and other partners evaluate different machine learning methods for aerodynamic analysis, uncertainty quantification and propagation in the context of data-driven aerodynamic modelling. The dataset consists of different CFD simulations, which were competed using the TAU solver for two different Reynolds numbers and (Re = 2.5×10^7 and 4×10^7). The flight condition parameters swept the whole envelope of the proposed aircraft, ranging the values of the Mach number from 0.5 to 0.95, and computing the polar for angles of attack spanning from -12^o to 15^o.

2 REVIEW OF THE STATE-OF-THE-ART

The following review of the state-of-the-art focuses in the use of machine learning techniques for the prediction of aerodynamic features in the aeronautical sector.

In the last five years, there have been several publications in this topic. Some of them have focused on the application of random forest models to predict the high-fidelity Reynolds-averaged Navier-Stokes (RANS) flow field, namely, the pressure, velocity, and turbulent viscosity [2] [5]. Others have researched in the use of local ML methods, as for instance in [6], where it is proposed a so-called Local Decomposition Method that makes use of machine learning tools to group the solutions of the training sample into homogeneous clusters and then build a model for each region.

The use of machine learning models to predict unsteady aerodynamics features has been also an interesting topic, regarding the recent publications. For instance, in [7] authors propose an unsteady aerodynamics and dynamic stall model using a long short-term memory variant of recurrent neural networks. The developed ML-based model was able to capture the key physics associated with dynamic stall, such as the precedence of moment stall before lift stall and cycle-to-cycle variations in the aerodynamic response. In addition, in [8], a machine-learning model for modelling time-dependent dynamics is used to construct an unsteady aerodynamic reduced-order model, which considers the variation of the airfoil geometry.

Given the amount of data that it is obtained from different data sources, such as simulations, the applicability of reduce order modelling (ROM) techniques are highly implemented on different areas such as on improving the performance of CFD [9], or in machine learning to speed up the fitting process of the models implemented and to reduce the computational cost [1]. In [10] is proposed the use of one of the ROM techniques called principal component analysis (PCA) for the structural optimization in different aerodynamic conditions using finite elements (FE). The proper orthogonal decomposition (POD) is also presented as another option in [11] to use along with the implementation of a surrogate model, based on steady turbulent aerodynamic fields in different conditions, to predict coefficients at diverse velocities. In

nonlinear structures, Isomap is introduced as a solution technique that tries to obtain an approximated low-dimensional nonlinear manifold solution with its associated local invers mapping process, for example, in [12] Isomap is developed to obtain an accurate prediction of shocks for different airfoils, while comparing it against the POD and the full-order CFD model.

Advanced neural networks have been also applied to this purpose, as for instance, in [13] where a deep learning model, named Dual Convolutional Neural Network (Dual-CNN) is developed and applied for the aero-engine turbines, or in [14] where a model for prediction of multiple aerodynamic coefficients of airfoils based on a convolutional neural network was proposed.

Finally, application of ML methods in the aerodynamic design optimization process has been also the subject of very recent publications [15] [16] [17] [18]. In particular, [15] provides an extensive review of ML applications contributing to aerodynamic design optimization from three fundamental perspectives: compact geometric design space, fast aerodynamic analysis, and efficient optimization architecture.

3 DESCRIPTION OF THE METHODS AND TOOLS

In this section, the different methods that will be applied in this paper are briefly introduced. References to the details and mathematical formulation of each of the methods are provided.

3.1 CART (Classification And Regression Trees)

Classification and regression trees algorithms, initially developed by Breiman et al. [19] consists of a process of nested decision rules, where observed data pass through each decision, from a prediction space, into another prediction space following a top-down, greedy approach through a binary splitting process, to obtain a final value that enters in the output space, determined by the algorithm stopping criterion defined.

Different metrics can be used to determine the best split, such as the squared error or the absolute error from the prediction statement as well as a maximum threshold to set the length of a subtree and prevent a poor prediction through the test phase.

Currently, regression tree algorithms are grouped inside the CART algorithms set. While a regression tree algorithm is used to predict a numerical label, a classification tree belongs for categorical data.

3.2 Random Forest

The use of decision trees for data prediction on big datasets can cause the model to be overfitted on the process; this means that the model only learned the association locally using a greedy algorithm that tries to minimize the error on each split. A solution for that can be the implementation of bagging technique, described by Breiman et al. [20], which randomly selects the number of predictors spaces for each tree created to reduce the high variance that a single decision tree can contain while calculating the accuracy to be the best. The use of bagging technique can cause correlation between some trees and as a result, the prediction could be erroneous.

Random forest technique [21] comes out as another option that tries to reduce this correlation while using the bagging technique. Instead of using the same amount of data to be considered

on each split of each tree, random samples of data are taken on each one, which increment the variation that each decision tree contains and reduce the correlation of each one.

3.3 POD

The proper orthogonal decomposition (POD) is a numerical method, typically implemented to be able to solve Navier-Stokes equations [22], which aims at obtaining a low-dimensional linear basis from the description approximation of a high-dimensional process.

The POD is a model reduction technique that allow identifying a set of POD modes that correlates between the original data and the new orthonormal basis vectors limited by that number of modes. The POD contains several interpretations [23]. The first interpretation uses the POD as the Karhunen-Loève decomposition (KLD); the other interpretation includes the use of KLD, the principal component analysis (PCA) [24], and the singular value decomposition (SVD). In this article, the SVD has been used among other methods for a data analysis approach.

SVD factorize a matrix A of type MxN into three matrices shown in equation *(1)*.

$$A = U\Sigma V^T \tag{1}$$

Where U *is* denoted as $[u_1, u_2, \dots u_m]$, that refers to the left-POD modes, is an MxN matrix of the orthonormal eigenvectors of AA^T, V^T , V is denoted as $[v_1, v_2, \dots v_n]$, that refers to the right-POD modes, is the transpose of an NxN matrix containing the orthonormal eigenvectors of A^TA and Σ is an NxN diagonal matrix of the singular values denoted in (*2*, which are the square roots of the eigenvalues of A^TA.

$$\Sigma = \begin{bmatrix} \sigma^1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma^n \end{bmatrix} \tag{2}$$

One of the key properties of the SVD implemented in this work is the possibility to approximate each one to specific low rank matrices that can still represent the same information [25]**.** This truncation technique consist on choosing a reduced number of modes given the rank of the Σ matrix, so that the most energetic ones are kept for its reconstruction and therefore, the most relevant features are conserved. In this work, the threshold to stablish how many modes was set to 99%

3.4 Optuna

Optuna [25] is a python library that provides several functionalities for optimization of the hyperparameters, which can be applied not also on a function of type *f(x)*, but also for hyperparameter tunning on a machine-learning algorithm to obtain the best configuration.

Optuna uses a technique called Sequential model-based optimization (SMBO or Bayesian optimization) [26] which is used for the evaluation of different parameters associated on problems that requires intensive computations for real physical problems like tunnel prediction modeling [27] or the prediction of axial compressors [28] or for aircraft design [29].

SMBO algorithm iterates over a surrogate function, which in this article uses the Tree Parzen Estimator (TPE) [30], to optimize the parameters selected previously on the target function. This process will require to quantify it over a custom defined metric through a Bayesian machine learning technique, usually a Gaussian regression, to narrow down the current space

of parameters, defined in this surrogated function, that the target can take, as shown in Figure 1.

1: Place a Gaussian process prior on f
2: Observe f at n_0 points according to an initial space-filling experimental design. Set $n = n_0$
3: **while** $n \leq N$ **do**
4: Update the posterior probability distribution on f using all available data
5: Let x_n be a maximizer of the acquisition function over x, where the acquisition function is computed using the current posterior distribution
6: Observe $y_n = f(x_n)$
7: Increment n
8: **end while**
9: Return a solution: either the point evaluated with the largest $f(x)$, or the point with the largets posterior mean

Figure 1: Basic Pseudo-code for Bayesian optimization taken from [26]

4 METHODOLOGY

A global approach has been followed through the training, test, and validation process on the database, were the 3D coordinates associated to the mesh nodes are not taken in consideration as values to train the models, however, the data is sorted by CaseID and by origin coordinate system of each configuration, which allows to speed up the model fitting part. Once the database is pre-processed, the optuna module is used to find the optimal sub-space of hyperparameters based on the custom loss function defined in equation ***(1)*** through 500 evaluations for the Decision Tree Regressor and 500 evaluations for the Random Forest algorithm (with and without using the ROM model).

For the POD model, hyperoptimization cannot be implemented intrinsically, however the number of POD-modes has been optimized from a range of it and where the minimum value is the number of modes is the one that keeps at least the 99% of information.

The space of hyperparameters declared for the optimization can be quite large into a point where the number of combinations can be made up to ~20 million and random forest optimization can take a lot of time, as hundreds of trees of different depth are built. In order to fasten the search process, a reduce space of hyperparameters have been selected.

The evaluation process of the models usually contains a training part, where train/test data is used to make the model learn from input and output data, and a validation data, where the model will predict output data from input data that the model have not previously known. In this case, instead of dividing the train-test and validation process independently, the validation data is appended to the test data as another entry in the data to be tested, afterwards, the metrics are also calculated for each case, which are shown in equation 2.

$$L(\hat{y}, y) = 0.5MAE(y, \hat{y}) + 0.5\left(1 - R^2(y, \hat{y})\right) \tag{2}$$

After all the models have been built, based on the metrics obtained, the best-fitted models results are selected to visualize them in terms of real vs predicted coefficient data and *X/c* vs predicted coefficient data.

A diagram of the aforementioned process is shown in Figure 2. It consists of two parts; the first one contains the model creation for the different configurations over the sub-space optimization founded with the optuna module on tree-based algorithms. The SVD technique is also applied on both cases for the evaluation of the models on the next part. In the second part, the r-squared, mean absolute error, mean squared error, maximum error and EVS metrics are used to evaluate each model for the POD and no POD processes to select the best and use them in the validation process, where validation data were extracted from each dataset previously.

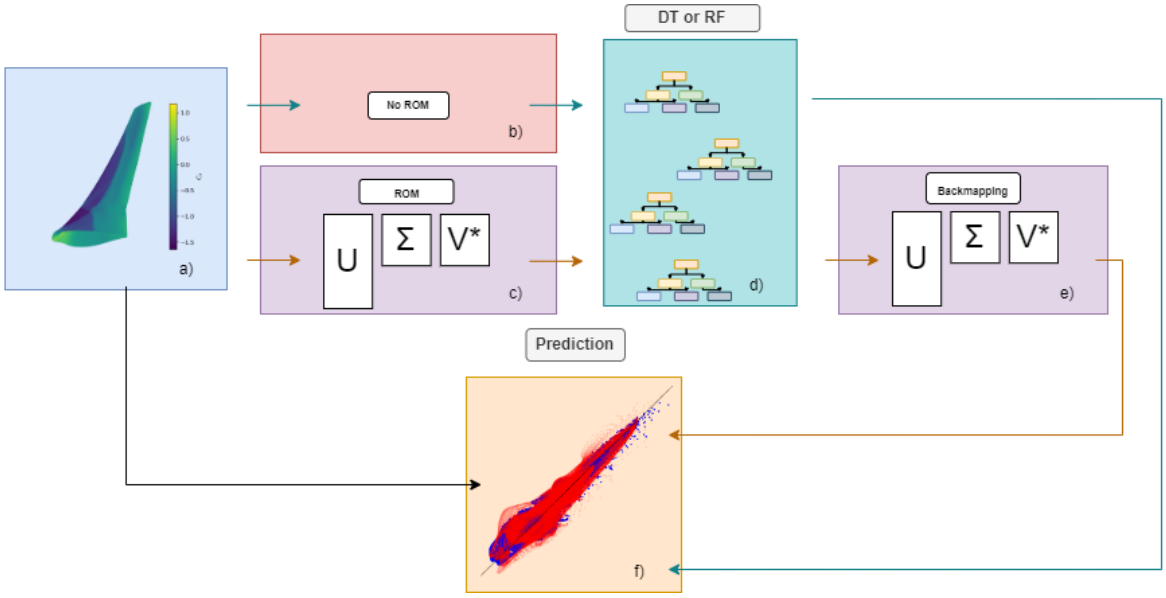

Figure 2: Diagram overview of the methodology: (a) Input data on the left. (b-c) no ROM model block and ROM model. (e) As the backmapping process. (d) Regression models implemented in the middle. (f) Output data in low middle block.

Two datasets ("2p5g_Sting_on_HTPoffVTPon_SST" and "m1g_Sting_On_HTPoffVTPon_SST") were generated from the same turbulence model and same mesh under different Mach and Alpha conditions, this allows to join both into a single dataset for model evaluations.

Table 1: Configurations used for model comparisons.

2p5g_Sting_on_HTPoffVTPon_SST m1g_Sting_On_HTPoffVTPon_SST	Datasets with CFD data of the XRF1 with sting and VTP, using SST turbulence model.
No_Sting_HTPoffVTPon_EARSM	Dataset with CFD data of the XRF1 without sting and with VTP, using EARSM turbulence model.

5 NUMERICAL RESULTS

Table 2 shows the results for the best combination of hyperparameters for the minimum loss function, regarding the model and the optuna module in terms of metrics for tree-based algorithms without ROM. In the table, the metrics calculated for the 500 iterations of the random forest were not possible, as the computational resources needed were higher than expected, instead, the grid search methodology was performed over a reduced number of variables for each hyperparameter. As can be observed, on the test dataset, the decision tree performs better for all configurations, while the random forest metrics were lower. This issue is probably explained by the fact that the decision tree could have overfit the data. To investigate it, the k-fold technique has been implemented, whose results are displayed on Table 3. The disperse range of results along all 5 k-fold might indicate that even with a meshless approach, overfitting can happen. On the other side, the limited range of values for the random forest process shows that more iterations could improve the metrics for the model generated.

Table 2: Metrics for each model on which combinations of type of algorithm hand hyperparameters are optimized.

Configuration	Model	Dataset	R^2	MAE	MSE	RMSE	Maximum error	EVS
2p5g_Sting_on_HTPoffVTPon_SST and m1g_Sting_On_HTPoffVTPon_SST	Decision tree	Test	0.937	0.056	0.0168	0.1004	1.324	0.929
		Validation	0.949	0.054	0.021	0.1089	1.373	0.914
	Random forest	Test	0.916	0.076	0.024	0.1415	1.411	0.887
		Validation	0.917	0.078	0.026	0.1454	1.408	0.885
No_Sting_HTPoffVTPon_EARSM	Decision tree	Test	0.925	0.068	0.020	0.1143	1.531	0.927
		Validation	0.901	0.080	0.032	0.1373	1.833	0.902
	Random forest	Test	0.904	0.098	0.026	0.1448	1.328	0.907
		Validation	0.874	0.126	0.037	0.1736	1.488	0.876

Table 3: Metrics for the first two configuration on a 5 k-fold with decision tree algorithm.

K-fold	Dataset	R^2	MAE	MSE	RMSE	Maximum error	EVS
1	Test	0.7856	0.1215	0.0565	0.200	2.3142	0.7881
	Validation	0.8928	0.0720	0.0277	0.1222	1.7839	0.8938
2	Test	0.8918	0.0788	0.0182	0.1235	1.1649	0.8938
	Validation	0.9167	0.0624	0.0194	0.1091	1.7559	0.9191
3	Test	0.8405	0.1026	0.0402	0.1718	2.0796	0.8489
	Validation	0.9120	0.0658	0.0220	0.1129	1.7733	0.9152
4	Test	0.7891	0.1082	0.0384	0.1633	2.3070	0.7932
	Validation	0.8809	0.0711	0.0255	0.1186	2.1435	0.8827
5	Test	0.8194	0.1137	0.0412	0.1670	1.15994	0.8505
	Validation	0.8911	0.0787	0.0285	0.1309	1.83593	0.9009

The optimal hyperparameters calculated for the best model are shown in Table 4. Because the three configurations are made for the XRF1 wing profile, the hyperparameters for each dataset should not tend to vary too much as well as their metrics, for that, their values have remain the same for each model.

Table 4: Hyperparameters associated to the best models for each dataset from **Table 2**

Configuration	Model	Max depth	Min samples split	Min samples leaf	Estimators	Bootstrap
2p5g_Sting_on_HTPoffVTPon_SST m1g_Sting_On_HTPoffVTPon_SST No_Sting_HTPoffVTPon_EARSM	Random forest	4	3	2	400	True

Table **5** shows the metrics for the best combination of hyperparameters for the minimum value of the loss function after 500 iterations on the random forest model with POD.

In comparison to Table 2, the metrics obtained reveal that the application of the truncated POD technique, reduces the precision of each model due to the loss of information, however, given the mathematical complexity that the POD technique require, the metrics are reasonable enough to indicate that the POD and, by extension, any ROM technique could be included together with machine learning algorithms.

Table 5: Metrics for each model on which the combinations of type of algorithm and use of truncation have

been made for the configurations data whose hyperparameters have been optimized.

Configuration	Model	Dataset	R^2	MAE	MSE	RMSE	Maximum error	EVS
2p5g_Sting_on_HTPoffVTPon_SST and m1g_Sting_On_HTPoffVTPon_SST	Random forest	Test	0.8876	0.0781	0.0259	0.1295	1.7053	0.8897
		Validation	0.8925	0.0748	0.0234	0.1215	1.8036	0.8950
No_Sting_HTPoffVTPon_EARSM	Random forest	Test	0.9486	0.0552	0.0148	0.0937	1.1692	0.9499
		Validation	0.9041	0.9041	0.0322	0.1353	1.6515	0.9061

The optimal hyperparameters calculated for the best models are shown in

Table ~~6~~. As well as in Table 2, the hyperparameters for all three configurations do not differ between each other.

Table 6: Hyperparameters associated to the best models for each dataset from TABLE3

Configuration	Model	nPODModes	Max depth	Min samples split	Min samples leaf	Estimators	Bootstrap
2p5g_Sting_on_HTPoffVTPon_SST m1g_Sting_On_HTPoffVTPon_SST No_Sting_HTPoffVTPon_EARSM	Random Forest	371	15	6	3	300	True

Even though different configurations have been tested, for the sake simplicity, in this manuscript, only the representation associated to the “2p5g_Sting_on_HTPoffVTPon_SST” configuration is shown.

Figure 3 shows a comparison of the pressure coefficient (*Cp*) for the validation case with Mach and Alpha values of 0.9040 and 6 respectively. In a), b) and c) random forest with POD correlates better than the random forest without POD. Even for both predictions, they are no able to predict accurately near the root and kink, which can be seen at e) and f) error distribution figures.

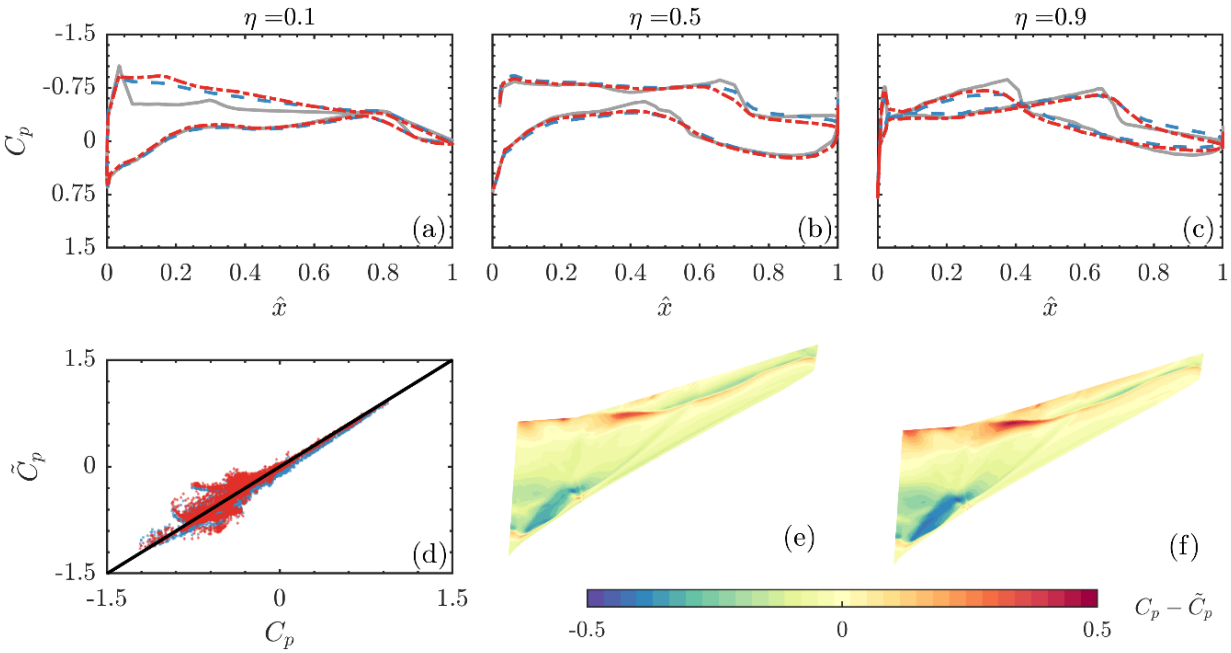

Figure 3: (a-c) *Cp* distribution for Mach of 0.9040 and Alpha 6 at wing sections 10%, 50% and 90%. Solid grey line for input *Cp*, short dashed red line for *Cp* from random forest and large dashed blue line for *Cp* from the random forest with POD. (d) Error regression representation between *Cp*'s predicted. (e,f) prediction error for the Cp distribution for random forest with POD(e) and random forest without POD (f).

Figure 4 shows a comparison of the pressure coefficient (*Cp*) for the validation case with a Mach and Alpha values of 0.8840 and 9.75 respectively. In this case, random forest with POD performs better at n = 50 and n = 90 (b and c). These results can be seen in the error distribution at e) and f), where the root prediction is worse with the random forest with POD model.

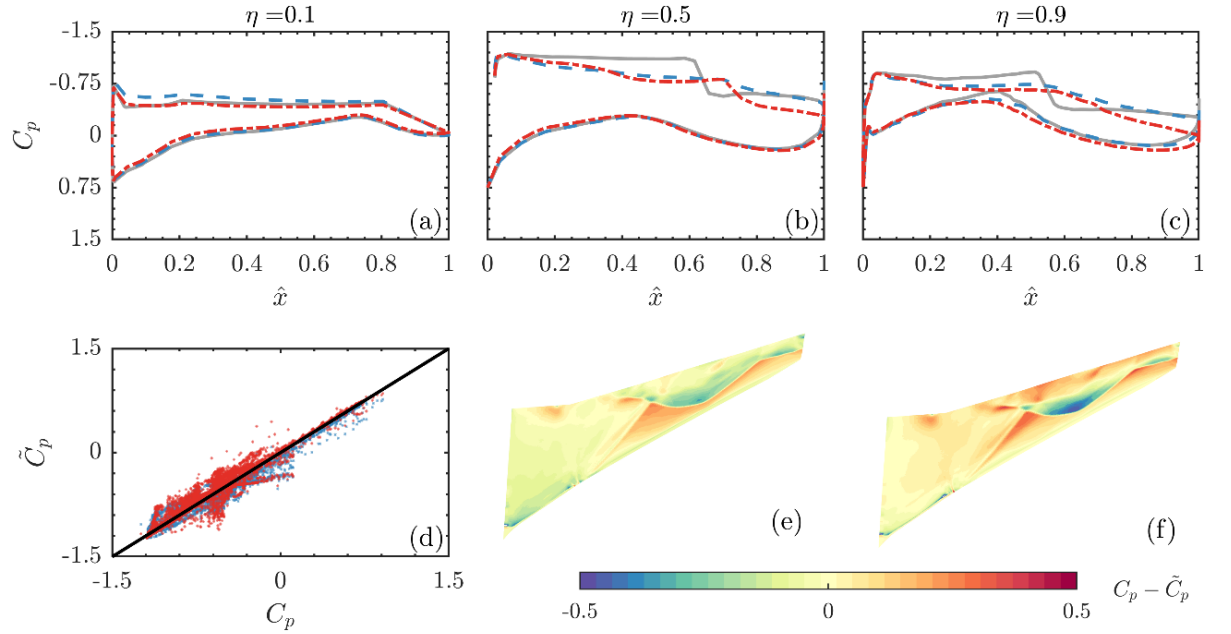

Figure 4: (a-c) *Cp* distribution for Mach of 0.8840 and Alpha 9.75 at wing sections 10%, 50% and 90%. Solid grey line for input *Cp*, short dashed red line for *Cp* from random forest and long dashed blue line for *Cp* from the random forest with POD. (d) Error regression representation between *Cp*'s predicted. (e,f) prediction error for the *Cp* distribution for random forest with POD (e) and random forest without POD (f).

7 CONCLUSIONS AND FUTURE WORK

This paper describes a study of different tree-based algorithms for the prediction of the pressure coefficient values using a global approach on the wing surface of the XRF1 dataset. The approach consists of two steps. In the first step, the data is pre-processed and the optimized hyperparameters are searched by using the optuna module for the decision tree and random forest algorithms with and without using the ROM model. In the second step, once the hyperparameters are found, the metrics are calculated for the testing data and validation data. After the whole process is done, the results reveal that the application of the random forest algorithm for each configuration obtains better results in terms of the metrics stablished.

Even though the models generated without using ROM modelling show better metrics, the complexity and amount of data that each tree needs to manipulate without dimensional reduction, make the possibility that each model generates overfitting, which means that its prediction results are only adjusted to the training data and do not generalize well for other dataset. For that, the models trained with the ROM method that appear on
Table 5 are selected to be the most reliable among all.

The surface along the wing does not follow a linear regime, as some areas are affected distinctly under different conditions, and therefore other reduction techniques that works under non-linear environments, such as Isomaps, could be implemented. Another approach to follow along with the ROM modelling refers to neural networks (NN). Given the amount of data that each configuration has, NN can be a solution to manage them properly and create models that overcome the previous models generated. Graph or convolutional neural networks are options to take into account, where the mesh as well as the connection between coordinates are also considered in the machine-learning model.

REFERENCES

[1] B. R. N. P. K. Steven L. Brunton, «Machine Learning for Fluid Mechanics,» *Annual Review of Fluid Mechanics,* vol. 52, nº 1, pp. 477-508, 2020.

[2] J. a. L. L. Nagawkar, «Multifidelity aerodynamic flow field prediction using random forest-based machine learning.,» *Aerospace Science and Technology,* 2022.

[3] W. Z. J. K. Y. L. Linyang Zhu, «Machine learning methods for turbulence modeling in subsonic flows around airfoils,» *Physics of Fluids,* vol. 31, nº 1, p. 015105, 2019.

[4] ML4AERO, «GARTEUR AD/AG-60 Machine learning and data-driven approaches for aerodynamic analysis and uncertainty quantification».

[5] A. a. A. K. G. Kumar, «Decision tree–and random forest–based novel unsteady aerodynamics modeling using flight data.,» *Journal of Aircraft,* vol. 56, nº 1, pp. 403-409, 2019.

[6] R. J.-C. J. a. P. S. Dupuis, «Aerodynamic data predictions for transonic flows via a machine-learning-based surrogate model.,» *AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference.,* 2018.

[7] G. e. a. Vijayakumar, «Enhancement of unsteady and 3D aerodynamics models using machine learning.,» *Journal of Physics: Conference Series.,* vol. 1452, nº 1, 2020.

[8] K. J. K. a. W. Z. Li, «Unsteady aerodynamic reduced-order modeling based on machine learning across multiple airfoils.,» *Aerospace Science and Technology,* vol. 119, nº 107173, 2021.

[9] M. V. M. K. N. e. a. Ripepi, «Reduced-order models for aerodynamic applications, loads and MDO,» *CEAS Aeronautical Journal ,* nº 9, p. 171–193, 2018.

[10] M. T. B. D. B. C. &. D. B. R. Bordogna, «Surrogate-based aerodynamics for composite wing box sizing,» *International Forum on Aeroelasticity and Structural Dynamics,* 2017.

[11] P. L. a. J. Alonso, «Investigation of non-linear projection for POD based reduced order models for Aerodynamics,» *39th Aerospace Sciences Meeting and Exhibit,* 2001.

[12] R. Z. S. G. &. N. K. T. Franz, «Interpolation-based reduced-order modelling for,» *International Journal of Computational Fluid Dynamics,* vol. 28, pp. 106-121, 2014.

[13] Y. e. a. Wang, «Dual-convolutional neural network based aerodynamic prediction and multi-objective optimization of a compact turbine rotor.,» *Aerospace Science and Technology,* vol. 116, nº 106869, 2021.

[14] H. e. a. Chen, «Multiple aerodynamic coefficient prediction of airfoils using a convolutional neural network.,» *Symmetry,* vol. 12, nº 4, p. 544, 2020.

[15] J. X. D. a. J. R. M. Li, «Machine Learning in Aerodynamic Shape Optimization.,» *arXiv preprint,* 2022.

[16] X. e. a. Yan, «Aerodynamic shape optimization using a novel optimizer based on machine learning techniques.,» *Aerospace Science and Technology,* vol. 86, pp. 826-835, 2019.

[17] J. e. a. Li, «Efficient aerodynamic shape optimization with deep-learning-based geometric filtering.,» *AIAA Journal,* vol. 58, nº 10, pp. 4243-4259, 2020.

[18] K. e. a. Balla, «An application of neural networks to the prediction of aerodynamic coefficients of aerofoils and wings.,» *Applied Mathematical Modelling,* vol. 96, pp. 456-479, 2021.

[19] L. F. J. O. R. &. S. C. Breiman, «Classification And Regression Trees,» *Routledge.,* vol. 1, 1984.

[20] L. Breiman, «Bagging Predictors,» *Machine Learning,* nº 24, p. 123–140, 1996.

[21] L. Breiman, «Random Forests,» *Machine Learning,* vol. 45, pp. 5-32, 2001.

[22] P. H. a. a. J. L. L. G Berkooz, «The Proper Orthogonal Decomposition in the Analysis of Turbulent Flows,» *Annual Review of Fluid Mechanics,* vol. 25, nº 1, pp. 539-575 , 1993.

[23] H. L. S. L. W. L. K. L. C. W. Y.C. LIANG, «PROPER ORTHOGONAL DECOMPOSITION AND ITS APPLICATIONS—PART I: THEORY,» *Journal of Sound and Vibration,* vol. 252, nº 3, pp. 527-544, 2002.

[24] I. Jolliffe, « Principal Component Analysis,» 2005.

[25] T. S. S. Y. T. O. T. &. K. M. Akiba, «Optuna: A Next-generation Hyperparameter Optimization Framework.,» *arXiv.,* 2019.

[26] P. I. Frazier, «A Tutorial on Bayesian Optimization,» *arXiv,* 2018.

[27] Y. L. Q. H. X. &. P. Y. Bo, «Real-time hard-rock tunnel prediction model for rock mass classification using CatBoost integrated with Sequential Model-Based Optimization.,» *Tunnelling and Underground Space Technology,* vol. 124, p. 104448., 2022.

[28] K. D. G. W. Y. &. N. J. Zhou, «Aeroheating and aerodynamic performance of a transonic hyperloop pod with radial gap and axial channel: A contrastive study.,» *Journal of Wind Engineering and Industrial Aerodynamics,* vol. 212, p. 104591, 2021.

[29] R. B. N. D. Y. &. S. A. Priem, «Upper trust bound feasibility criterion for mixed constrained Bayesian optimization with application to aircraft design.,» *Aerospace Science and Technology,* vol. 105, p. 105980, 2020.

[30] J. &. B. R. &. K. B. &. B. Y. Bergstra, «Algorithms for Hyper-Parameter Optimization.,» *Advances in Neural Information Processing Systems,* 2011.

[31] P. Hansen, «The truncatedSVD as a method for regularization.,» *BIT ,* nº 27, p. 534–553 , 1987.