

**INDIAN INSTITUTE OF TECHNOLOGY HYDERABAD**  
Department of Physics  
Kandi, Telangana, India – 502285



### **Project-1 (EP3105)**

A Project Report Submitted in Partial Fulfilment of the Requirements  
for the Degree of Bachelor of Technology in Physics

## **Airfoil's Aerodynamic Coefficient Prediction using Artificial Neural Networks**

**Submitted by:**

**NEVIL C PRINCE**  
**EP23BTECH11019**

**Under the Supervision of:**

**Dr. Anupam Gupta**  
Department of Physics, IIT Hyderabad

**Dr. Lakshmana Dora Chandrala**  
Department of Mechanical & Aerospace Engineering, IIT Hyderabad

**July 2025 – December 2025**

# Acknowledgement

I would like to express my deep appreciation to my supervisors, Dr. Anupam Gupta of the Department of Physics and Dr. Lakshmana Dora Chandrala of the Department of Mechanical and Aerospace Engineering, for their expert guidance, constructive feedback, and continued support throughout the development of this project. Their academic insight and mentorship have been invaluable in shaping the direction and quality of this work.

I would also like to acknowledge the Department of Physics and the Department of Mechanical and Aerospace Engineering at the Indian Institute of Technology Hyderabad for providing the academic platform that enabled this project. I am equally grateful for the academic structure and learning opportunities that have contributed to my development during this period.

# Abstract

Selecting an appropriate airfoil profile is an essential step in the early phase of aircraft or UAV design, as its geometry directly influences the resulting aerodynamic performance. The aerodynamic coefficients associated with an airfoil—namely lift, drag and pitching moment—play a key role not only in evaluating aerodynamic efficiency but also in designing control systems and predicting aeroelastic behaviour. Traditionally, these coefficients are determined either through wind-tunnel experiments or numerical methods such as CFD, both of which can be time-consuming and computationally expensive, especially when exploring large design spaces.

In this project, a data-driven alternative is explored using Artificial Neural Networks (ANNs) to predict the aerodynamic coefficients of NACA airfoils across varying geometric configurations and flight conditions, including angle of attack, Reynolds number and Mach number. The datasets were generated using JavaFoil, and multiple neural network configurations were trained and evaluated to identify an architecture capable of capturing nonlinear aerodynamic trends with reasonable accuracy. The developed model demonstrates that ANNs can effectively learn the relationship between airfoil geometry and aerodynamic response, offering a faster way to estimate coefficients compared to conventional simulation-based approaches. The results highlight the potential of machine learning-based surrogate models in preliminary aerodynamic design workflows, where rapid estimation is valuable.

# Table of Contents

<b>Acknowledgement</b> .....	i
<b>Abstract</b> .....	ii
<b>I.</b> <b>Introduction</b> .....	1
<b>II.</b> <b>Theoretical Background</b> .....	2
Airfoil & its Aerodynamics .....	2
Artificial Neural Networks .....	4
<b>III.</b> <b>Methodology</b> .....	6
Data Preparation .....	6
Data Preprocessing .....	7
Network Architecture .....	9
<b>IV.</b> <b>Results &amp; Discussion</b> .....	10
Network Performance due to Multiple Hidden Layers .....	10
Network Performance due to Increasing Neurons .....	12
Network Performance due to Varying Geometric Information .....	13
Network Performance on Different Datasets .....	14
<b>V.</b> <b>Conclusion</b> .....	15
<b>References</b> .....	16

# I. Introduction

Airfoil geometry plays a central role in the aerodynamic performance of any aircraft or rotorcraft. When an airfoil moves through air, it experiences aerodynamic forces and moment that depend strongly on its shape. The lift force acts perpendicular to the incoming flow, while drag acts parallel to it, resisting motion. During the design process, engineers rely on these aerodynamic characteristics—such as lift, drag, moment coefficients, aerodynamic center and center of pressure—to estimate the performance of a complete wing or vehicle. Because these properties vary significantly with airfoil geometry, selecting an optimal airfoil shape remains a major task in aircraft design.

With increased computing capability, engineers are now able to evaluate aerodynamic characteristics more efficiently. In addition to traditional computational methods like CFD, the growing influence of data-driven techniques has opened new possibilities for modeling complex aerodynamic behavior. Among these, Artificial Neural Networks (ANNs) have emerged as powerful tools capable of learning nonlinear mappings directly from data. Their ability to infer relationships between geometry, flow conditions, and resulting aerodynamic coefficients makes them appealing for early-stage design and optimization, especially when speed is a priority.

While CFD is grounded in physical laws and provides detailed flow solutions, it can be computationally heavy and unsuitable for large parametric studies or applications requiring near real-time predictions. In contrast, ANNs offer rapid prediction after training, without solving fluid equations each time. This makes them attractive for tasks such as flight control, inverse design, and system identification where instantaneous aerodynamic feedback may be needed.

Recent developments in machine learning for fluid mechanics demonstrate that deep learning approaches—particularly those incorporating convolutional structures—can learn lift, drag, pressure fields and flow behavior around airfoils from data. These advances show that neural networks can replace or augment traditional aerodynamic analysis in certain regimes. Beyond flow prediction, neural network-based methods have also been used to support inverse design, optimization of airfoil geometry, and improved understanding of aerodynamic limits such as stall.

In this context, this project focuses on predicting the aerodynamic coefficients of NACA airfoil shapes using neural networks. Instead of relying on predefined geometric parameters alone, the airfoil profiles are represented through normalized surface coordinate data, enabling the network to directly learn the design space of geometries. By training on a diverse dataset covering variations in angle of attack and flow conditions, the model aims to capture the aerodynamic trends across different airfoil configurations and provide fast, accurate coefficient predictions suitable for engineering applications.

## II. Theoretical Background

### Airfoil & its Aerodynamics

An airfoil can be defined as the cross-sectional shape of a wing or lifting surface. Its geometry plays a key role in determining aerodynamic behavior, as it directly influences the forces and moments generated during flight. Figure I shows the commonly used nomenclature associated with airfoils. The main terms are explained below:

- **Mean Camber Line:** The curve connecting the points midway between the upper and lower surfaces of the airfoil.
- **Leading Edge Radius:** The radius of a theoretical circle at the leading edge, centered on the mean camber line and tangent to both surfaces.
- **Leading Edge and Trailing Edge:** The forward-most and the rear-most edge of an airfoil, respectively.
- **Chord Line:** A straight line having length  $c$ , that connects the forward-most and rear-most points of an airfoil.
- **Camber:** The maximum perpendicular distance between the mean camber line and the chord line.
- **Thickness:** The maximum separation between the upper and lower surfaces at any chord-wise point.
- **Angle of Attack ( $\alpha$ ):** The angle between the free-stream airflow ( $V_\infty$ ) and the chord line.
- **Lift ( $L$ ):** The component of the aerodynamic force ( $R$ ) acting perpendicular to the free-stream flow.
- **Drag ( $D$ ):** The component of  $R$  acting parallel to the flow direction, opposing forward motion.
- **Moment ( $M$ ):** The aerodynamic tendency to rotate the airfoil about a selected reference point due to pressure and shear distribution.

The aerodynamic characteristics of an airfoil—particularly lift, drag, and moment—depend strongly on the mean camber line, the position of maximum camber, and the thickness distribution along the chord.

For standardization and ease of comparison, these quantities are expressed in non-dimensional form as aerodynamic coefficients:

$$C_L = \frac{L}{qc}, \quad C_D = \frac{D}{qc}, \quad C_m = \frac{M}{qc^2} \quad (\text{II.1})$$

where the dynamic pressure is defined as:

$$q = \frac{1}{2}\rho V_\infty^2 \quad (\text{II.2})$$

Extensive experimental work carried out by the National Advisory Committee for Aeronautics (NACA), later incorporated into NASA, resulted in standardized airfoil series. Among these, the NACA 4- and 5-digit families are widely used and relevant to this project.

The NACA 4-digit designation follows the convention:

1. The first digit specifies the maximum camber as a percentage of the chord.
2. The second digit indicates the chord-wise position of maximum camber in tenths of the chord.
3. The last two digits denote the maximum thickness of the airfoil as a percentage of the chord.

The more advanced NACA 5-digit format is written as **LPSTT**, where:

1.  $L$  represents the theoretical design lift coefficient.
2.  $P$  specifies the maximum camber position along the chord.
3.  $S$  indicates whether the camber is standard (0) or reflexed (1).
4.  $TT$  defines the maximum thickness as a percentage of chord.

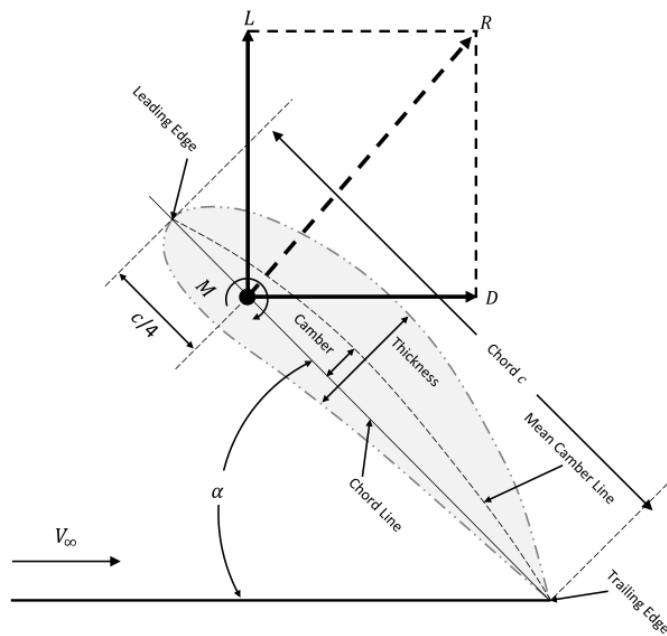


Figure I: Airfoil Nomenclature

## **Artificial Neural Networks**

An artificial neural network is conceptually comprised of a collection of nodes and connections. The basic elements of an artificial neural network are called neurons; they represent the sites that process information. The interconnecting links between the processing units, or neurons, are called synapses. Each synapse can be characterized by a weight, which is represented by a numerical value. All neurons in the adjacent layers are connected and the flow of information is restricted to the forward direction. The network consists of three layers: input layer, hidden layer(s) and output layer. The hidden layer enables the artificial neural network to learn relationships between input–output variables through suitable mappings. Among the many artificial neural network models, the backpropagation algorithm is one of the better known and frequently used.

Back propagation was created by generalizing the Widrow-Hoff learning rule to multiple-layer networks and nonlinear differentiable transfer functions. Input–output pairs are used to train an artificial neural network until it can approximate a function. Back propagation was the first practical method for training a multiple-layer feed forward network. An artificial neural network's initial weights are simply random numbers, which change during training. Training consists of presenting actual experimental data to the artificial neural network and using a mathematical algorithm—the back propagation algorithm—to adjust the weights. Each pair of patterns goes through two stages of activation: a forward pass and a backward pass. The forward pass involves presenting a sample input to the artificial neural network and letting the activations flow until they reach the output layer. During the backward pass, the network's actual output (from the forward pass) is compared with the target output and errors are computed for the output units. Adjustments of weights are based on the difference between the correct and computed outputs. Once each observation's computed and actual outputs are within the specified error tolerance, training stops and the artificial neural network is ready for use: given a new input observation, it will estimate what the corresponding output values should be. After extensive training, the artificial neural network eventually establishes the input–output relationships through the adjusted weights on the network.

Learning in an artificial neural network is typically accomplished using examples. This is also called “training” of the artificial neural network because the learning is achieved by adjusting the connection weights in the artificial neural network iteratively so that the trained (or learned) artificial neural network can perform certain tasks. Learning in an artificial neural network can roughly be divided into supervised, unsupervised, and reinforcement learning.

Supervised learning is based on direct comparison between the actual output of an artificial neural network and the desired correct output, also known as the target output. It is often formulated as the minimization of an error function such as the total mean square error between the actual output and the desired output summed over all available data. A gradient descent-based optimization algorithm such as backpropagation can then be used to adjust connection weights in the artificial neural network iteratively in order to minimize the error.

Reinforcement learning is a special case of supervised learning where the exact desired output is unknown. It is based only on the information of whether or not the actual output is correct.

Unsupervised learning is solely based on the correlations among input data. No information on “correct output” is available for learning. The essence of a learning algorithm is the learning rule, i.e., a weight-updating rule which determines how connection weights are changed. The success of the artificial neural network depends greatly on defining the influencing parameters for the problem, definition of suitable artificial neural network architecture and the learning algorithm.

## III. Methodology

This section explains the workflow used to acquire, clean, and transform the raw data before model training. This stage is critical, as it may include correcting errors, restructuring files, and merging separate datasets to improve the overall quality and diversity of training data.

### Data Preparation

A dataset of NACA 4- and 5-digit airfoils was generated using automated macro scripts in Javafoil. This automation allowed large-scale batch processing without manual input and helped maintain uniformity across all generated samples. Each airfoil shape was discretized using 101 cosine-spaced points along the unit chord, which provided smooth definitions of both the upper and lower surfaces. Along with the geometry, aerodynamic coefficients including lift ( $C_L$ ), drag ( $C_D$ ), and pitching moment ( $C_m$ ) were computed for a range of angles of attack (AoA), Reynolds numbers, and Mach numbers.

For all aerodynamic evaluations, CalcFoil was used as the stall modelling approach, while boundary layer transition was handled using the Drela  $e^n$  method (based on the post-1991 Xfoil transition model). These models were chosen as they offer a good balance between speed and accuracy for subsonic airfoil analysis.

After generating the data, the coordinates were interpolated to  $N$  fixed cosine-spaced locations along the  $x$ -axis (Fig. II), giving upper ( $y_{U,k}$ ) and lower ( $y_{L,k}$ ) surface values for  $k \in [1, N]$ . The cosine spacing ensured denser point placement near the leading and trailing edges, where the geometry changes most rapidly. Although the points at  $(0, 0)$  and  $(1, 0)$  were included during shape definition, they were excluded later since they were identical for every airfoil and did not contribute useful information.

In total, six datasets were created—three for the NACA 4-digit family and three for the 5-digit family—based on different parameter sweeps for thickness, camber, mean-line definition, and operating conditions. Table I lists the naming scheme and the corresponding ranges used. This process resulted in 560 unique 4-digit airfoils and 1120 unique 5-digit airfoils. With aerodynamic properties evaluated across 315 operating conditions for each geometry, the final dataset contained 176,400 samples for the 4-digit series and 352,800 samples for the 5-digit series.

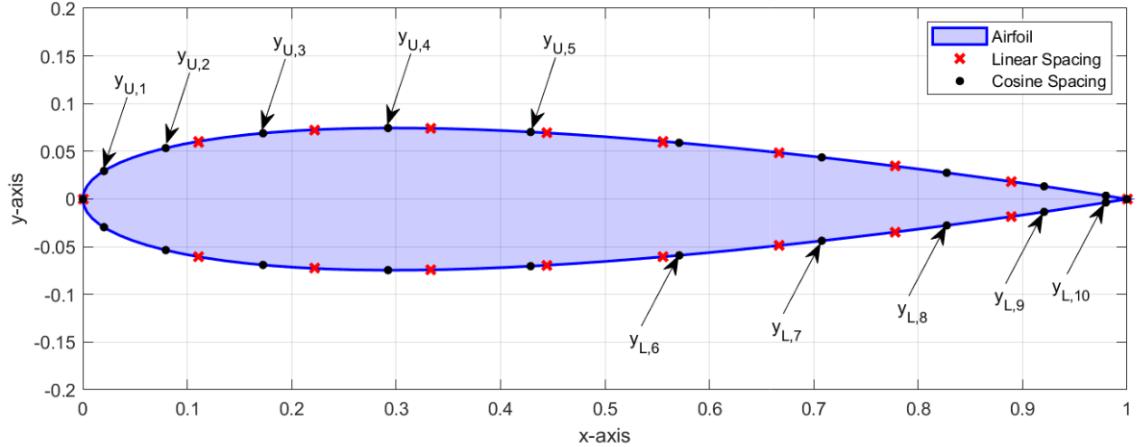


Figure II: Cosine Spacing at  $N = 10$

Table I: Ranges for Airfoil Parameters & Flight Conditions

No. of Coordinate Points ( $\mathcal{N}$ )		NACA 4-Digit Series			NACA 5-Digit Series		
5		NACA4D.05			NACA5D.05		
10		NACA4D.10			NACA5D.10		
15		NACA4D.15			NACA5D.15		
		Min	Max	Step	Min	Max	Step
Airfoil Parameters		Thickness (%)	5	35	5	5	35
		Location of Max. Thickness (%)			30 (fixed)		
		Max. Camber (%)			30 (fixed)		
		Design Lift Coefficient			Not Applicable		
		Location of Max. Camber (%)			0	1.8	0.2
Flight Conditions		Reflexed Trailing Edge			Both Standard & Reflexed		
		Angle of Attack (deg)			-10	10	1
		Reynolds No.			$1 \times 10^5$	$5 \times 10^5$	$1 \times 10^5$
		Mach No.			0.1	0.3	0.1

## Data Preprocessing

Once the datasets were generated, the next step was to clean and organize them before using them for neural network training. For the NACA 4-digit datasets, each sample contained the thickness, camber, and position of maximum camber as the first three columns, followed by  $N$  points defining the upper surface and another  $N$  points defining the lower surface. The remaining six columns corresponded to AoA, Mach number, Reynolds number,  $C_L$ ,  $C_D$ , and  $C_m$  values.

For the NACA 5-digit datasets, the format was similar, except the first four columns included thickness, design lift coefficient ( $C_L$  design), position of maximum camber, and an identifier indicating whether the camber line was simple or reflex. The rest of the structure remained consistent with the 4-digit format.

During inspection, it was found that some of the 5-digit samples did not have the trailing edge lying on the  $x$ -axis. A tolerance of 0.015 (non-dimensionalized with chord length) was used

as a threshold to determine whether a sample was acceptable. All airfoils exceeding this tolerance were removed to avoid introducing geometric inconsistencies into the dataset. In addition, some 5-digit cases with very high camber near the edges resulted in Javafoil failing to compute aerodynamic coefficients reliably. These samples were also discarded. It is important to note that the NACA 4-digit series did not face these issues and all samples in that category were processed successfully.

After filtering and cleanup, the final dataset contained 176,400 valid samples for the 4-digit airfoils and 236,880 valid samples for the 5-digit airfoils. Each dataset was then loaded separately into individual Jupyter notebooks, resulting in six notebooks in total—one for each dataset.

Before preparing the data for training, the feature columns related to geometric parameters were removed. For the NACA 4-digit series, the first three columns (thickness, camber, and position of maximum camber) were discarded. For the NACA 5-digit series, the first four columns (thickness, design lift coefficient, position of maximum camber, and camber type: simple or reflex) were removed. The remaining columns consisted solely of the interpolated surface coordinates and aerodynamic coefficients.

Once formatted, each dataset was randomly shuffled to avoid any ordering bias and improve the stability and generalization capability of the learning process. The shuffled data was then split into training, validation, and testing subsets using a 70:15:15 ratio.

The input features and output target values were separated for each split. Finally, all  $(2N + 3)$  input features in the training set were standardized by subtracting their mean and scaling to unit variance. The same normalization parameters were then applied to the validation and testing sets to maintain consistency across all subsets. A schematic representation of the neural network structure, including the input and output mapping, is shown in Fig. III.

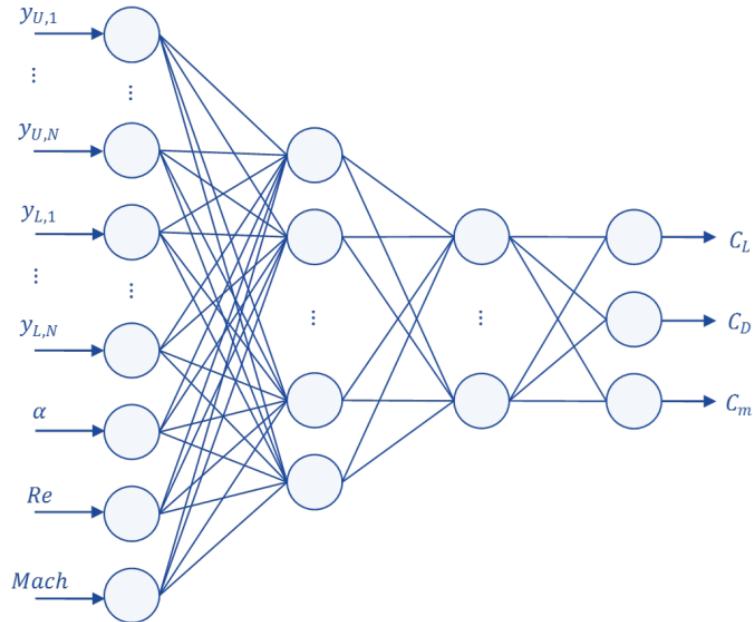


Figure III: Cosine Spacing at  $N = 10$

### Network Architecture

All neural network models were developed using the TensorFlow package with the Keras API in Python. The Adam optimizer was used for training, with an initial learning rate of 0.0005. The first and second moment decay rates were set to 0.9 and 0.999, respectively. Mean Squared Error (MSE) was used as the loss function, while Root Mean Squared Error (RMSE) and  $R^2$  were used as evaluation metrics. ReLU activation functions were applied in all hidden layers because of their simplicity and reliable performance in avoiding vanishing gradients.

Each model was trained for 50 epochs using mini-batches of 128 samples. During training, the validation loss was monitored, and the learning rate was reduced by 10% if no improvement was observed for five consecutive epochs (patience = 5).

# IV. Results & Discussion

A total of 20 training runs were performed for each case listed in Tables II-XIII. Across all runs, the training, validation, and testing losses showed consistent behavior, suggesting stable learning and good generalization within the first 10 epochs. Therefore, the RMSE and  $R^2$  values reported in the tables correspond to the model that achieved the best average test loss among all runs.

The purpose of running these iterative experiments was to identify a lightweight neural network architecture that delivered strong performance for all three aerodynamic outputs ( $C_L$ ,  $C_D$ , and  $C_m$ ). In general, networks that are wide but shallow tend to memorize patterns rather than generalize them well. On the other hand, deeper networks are typically better at capturing nonlinear behavior and learning meaningful features at multiple levels. Because of this, it was necessary to balance both the number of layers and neurons to find a suitable compromise between accuracy and computational cost.

## Network Performance due to Multiple Hidden Layers

The first group of experiments focused on identifying a suitable network depth. All six datasets were used in this stage, and the architecture followed a simple structure in which the number of neurons was reduced by half in each successive hidden layer. The final layer contained three neurons corresponding to the predicted aerodynamic coefficients. As it can be seen from the following tables, the network performance generally improved when increasing the depth from two to four hidden layers, with the four-layer configuration producing the strongest results across most datasets. Beyond this point, further increasing network depth did not provide meaningful improvements and, in some cases—such as the five-layer configuration—resulted in a slight reduction in performance.

Table II: NACA4D\_05

Case No.	Network Architecture	RMSE			$R^2$		
		$C_L$	$C_D$	$C_m$	$C_L$	$C_D$	$C_m$
1	64, 3	0.004378	0.004406	0.002037	0.999977	0.841861	0.999674
2	64, 32, 3	0.003103	0.004121	0.001734	0.999989	0.861625	0.999763
<b>3</b>	<b>64, 32, 16, 3</b>	<b>0.003109</b>	<b>0.004088</b>	<b>0.001695</b>	<b>0.999989</b>	<b>0.863868</b>	<b>0.999774</b>
4	64, 32, 16, 8, 3	0.003280	0.004120	0.001757	0.999987	0.861719	0.999756

Table III: NACA4D\_10

Case No.	Network Architecture	RMSE			$R^2$		
		$C_L$	$C_D$	$C_m$	$C_L$	$C_D$	$C_m$
1	64, 3	0.005312	0.004230	0.002296	0.999967	0.846132	0.999583
2	64, 32, 3	0.004231	0.003978	0.001881	0.999979	0.863597	0.999721
3	64, 32, 16, 3	0.004128	0.003866	0.001939	0.999980	0.871552	0.999701
<b>4</b>	<b>64, 32, 16, 8, 3</b>	<b>0.003975</b>	<b>0.003922</b>	<b>0.001817</b>	<b>0.999981</b>	<b>0.867454</b>	<b>0.999738</b>

Table IV: NACA4D\_15

Case No.	Network Architecture	RMSE			$R^2$		
		$C_L$	$C_D$	$C_m$	$C_L$	$C_D$	$C_m$
1	64, 3	0.006077	0.004670	0.002374	0.999956	0.821334	0.999555
2	64, 32, 3	0.004789	0.004284	0.002098	0.999973	0.850207	0.999651
<b>3</b>	<b>64, 32, 16, 3</b>	<b>0.004540</b>	<b>0.004256</b>	<b>0.001997</b>	<b>0.999976</b>	<b>0.852136</b>	<b>0.999686</b>
4	64, 32, 16, 8, 3	0.004478	0.004403	0.002031	0.999976	0.839429	0.999675

Table V: NACA5D\_05

Case No.	Network Architecture	RMSE			$R^2$		
		$C_L$	$C_D$	$C_m$	$C_L$	$C_D$	$C_m$
1	64, 3	0.005173	0.005346	0.002402	0.999969	0.758661	0.999329
2	64, 32, 3	0.003965	0.005013	0.002007	0.999982	0.787819	0.999533
3	64, 32, 16, 3	0.003970	0.005053	0.002073	0.999982	0.784293	0.999499
<b>4</b>	<b>64, 32, 16, 8, 3</b>	<b>0.003964</b>	<b>0.005033</b>	<b>0.002061</b>	<b>0.999981</b>	<b>0.785979</b>	<b>0.999508</b>

Table VI: NACA5D\_10

Case No.	Network Architecture	RMSE			$R^2$		
		$C_L$	$C_D$	$C_m$	$C_L$	$C_D$	$C_m$
1	64, 3	0.005329	0.005838	0.002324	0.999967	0.721204	0.999359
2	64, 32, 3	0.004234	0.005447	0.001917	0.999979	0.757878	0.999566
<b>3</b>	<b>64, 32, 16, 3</b>	<b>0.004475</b>	<b>0.005434</b>	<b>0.002041</b>	<b>0.999976</b>	<b>0.758979</b>	<b>0.999510</b>
4	64, 32, 16, 8, 3	0.004295	0.005438	0.001958	0.999978	0.758787	0.999549

Table VII: NACA5D\_15

Case No.	Network Architecture	RMSE			$R^2$		
		$C_L$	$C_D$	$C_m$	$C_L$	$C_D$	$C_m$
1	64, 3	0.005831	0.006601	0.002401	0.999960	0.670914	0.999297
2	64, 32, 3	0.004880	0.006318	0.002039	0.999972	0.698665	0.999493
<b>3</b>	<b>64, 32, 16, 3</b>	<b>0.004593</b>	<b>0.006250</b>	<b>0.001990</b>	<b>0.999975</b>	<b>0.705177</b>	<b>0.999519</b>
4	64, 32, 16, 8, 3	0.004282	0.006311	0.002023	0.999978	0.699260	0.999500

### Network Performance due to Increasing Neurons

The second phase of experiments focused on identifying an appropriate network width. Using the four-layer configuration from previous part, the number of neurons in each hidden layer was progressively doubled. Increasing the neuron count generally improved prediction accuracy; however, it also resulted in higher computational cost. The performance difference between Case 4 and Case 5 was relatively small and did not justify the additional complexity and training time. Therefore, Case 4 was considered adequate for the purposes of this study.

Table VIII: NACA4D\_05

Case No.	Network Architecture	RMSE			$R^2$		
		$C_L$	$C_D$	$C_m$	$C_L$	$C_D$	$C_m$
1	64, 32, 16, 3	0.003221	0.004070	0.001648	0.999988	0.865080	0.999786
2	128, 64, 32, 3	0.002847	0.003759	0.001304	0.999990	0.884898	0.999866
3	256, 128, 64, 3	0.002647	0.003550	0.001043	0.999991	0.897266	0.999913
4	512, 256, 128, 3	0.002013	0.003299	0.000912	0.999995	0.911276	0.999931
<b>5</b>	<b>1024, 512, 256, 3</b>	<b>0.002175</b>	<b>0.003238</b>	<b>0.000879</b>	<b>0.999994</b>	<b>0.914467</b>	<b>0.999935</b>

Table IX: NACA4D\_10

Case No.	Network Architecture	RMSE			$R^2$		
		$C_L$	$C_D$	$C_m$	$C_L$	$C_D$	$C_m$
1	64, 32, 16, 3	0.004142	0.003848	0.001800	0.999980	0.872731	0.999745
2	128, 64, 32, 3	0.003152	0.003521	0.001312	0.999888	0.893397	0.999864
3	256, 128, 64, 3	0.002699	0.003251	0.000991	0.999990	0.909062	0.999923
4	512, 256, 128, 3	0.002267	0.003147	0.000868	0.999994	0.914501	0.999940
<b>5</b>	<b>1024, 512, 256, 3</b>	<b>0.002266</b>	<b>0.003086</b>	<b>0.000864</b>	<b>0.999993</b>	<b>0.917466</b>	<b>0.999939</b>

Table X: NACA4D\_15

Case No.	Network Architecture	RMSE			$R^2$		
		$C_L$	$C_D$	$C_m$	$C_L$	$C_D$	$C_m$
1	64, 32, 16, 3	0.004882	0.004210	0.002079	0.999972	0.855346	0.999658
2	128, 64, 32, 3	0.003589	0.003887	0.001435	0.999985	0.876686	0.999837
3	256, 128, 64, 3	0.003168	0.003581	0.001131	0.999987	0.895340	0.999898
4	512, 256, 128, 3	0.002750	0.003425	0.000968	0.999990	0.904181	0.999922
<b>5</b>	<b>1024, 512, 256, 3</b>	<b>0.002573</b>	<b>0.003352</b>	<b>0.000895</b>	<b>0.999991</b>	<b>0.908223</b>	<b>0.999935</b>

Table XI: NACA5D\_05

Case No.	Network Architecture	RMSE			$R^2$		
		$C_L$	$C_D$	$C_m$	$C_L$	$C_D$	$C_m$
1	64, 32, 16, 3	0.003971	0.004966	0.002019	0.999982	0.791679	0.999527
2	128, 64, 32, 3	0.002889	0.004487	0.001356	0.999990	0.829923	0.999786
3	256, 128, 64, 3	0.002704	0.004186	0.001117	0.999991	0.851863	0.999852
<b>4</b>	<b>512, 256, 128, 3</b>	<b>0.002453</b>	<b>0.003878</b>	<b>0.000903</b>	<b>0.999993</b>	<b>0.872907</b>	<b>0.999905</b>
5	1024, 512, 256, 3	0.002256	0.004002	0.000952	0.999994	0.863891	0.999891

Table XII: NACA5D\_10

Case No.	Network Architecture	RMSE			$R^2$		
		$C_L$	$C_D$	$C_m$	$C_L$	$C_D$	$C_m$
1	64, 32, 16, 3	0.003988	0.005394	0.001973	0.999981	0.762696	0.999539
2	128, 64, 32, 3	0.003263	0.004895	0.001391	0.999987	0.804653	0.999772
3	256, 128, 64, 3	0.002760	0.004492	0.001151	0.999991	0.834945	0.999841
<b>4</b>	<b>512, 256, 128, 3</b>	<b>0.002316</b>	<b>0.004186</b>	<b>0.001028</b>	<b>0.999994</b>	<b>0.857025</b>	<b>0.999874</b>
5	1024, 512, 256, 3	0.002461	0.004189	0.000985	0.999992	0.856510	0.999883

Table XIII: NACA5D\_15

Case No.	Network Architecture	RMSE			$R^2$		
		$C_L$	$C_D$	$C_m$	$C_L$	$C_D$	$C_m$
1	64, 32, 16, 3	0.004566	0.006282	0.002029	0.999975	0.701907	0.999498
2	128, 64, 32, 3	0.003467	0.005818	0.001506	0.999986	0.744199	0.999724
3	256, 128, 64, 3	0.002889	0.005332	0.001147	0.999990	0.785196	0.999833
4	512, 256, 128, 3	0.002596	0.005229	0.000994	0.999991	0.792163	0.999879
<b>5</b>	<b>1024, 512, 256, 3</b>	<b>0.002564</b>	<b>0.005044</b>	<b>0.001025</b>	<b>0.999992</b>	<b>0.807782</b>	<b>0.999867</b>

### Network Performance due to Varying Geometric Information

In addition to evaluating network depth and width, the model performance was also tested by varying the amount of geometric information supplied as input. Unlike Computational Fluid

Dynamics (CFD), where mesh resolution and geometric detail have a significant impact on the accuracy of aerodynamic predictions, this experiment aimed to determine whether the artificial neural network could still learn aerodynamic behavior even when only limited surface coordinate data were provided.

As summarized in Table XIV, three datasets were compared: NACA4D\_05, NACA4D\_10, and NACA4D\_15, representing cases where the upper and lower surface coordinates were sampled at five, ten, and fifteen chordwise locations, respectively. For this comparison, the final network architecture selected from earlier experiments was used. The results showed that the model performed well across all three datasets. Even with the least geometric detail (NACA4D\_05), the network achieved  $R^2$  values greater than 0.90 for all outputs, indicating that it was still able to learn meaningful aerodynamic relationships from limited geometry information. However, the NACA4D\_10 dataset provided slightly better overall performance compared to NACA4D\_05 and NACA4D\_15, and therefore was selected as the final configuration for this project.

Table XIV: Network Performance due to Varying Geometric Information [512, 256, 128, 3]

Case No.	Dataset	RMSE			$R^2$		
		$C_L$	$C_D$	$C_m$	$C_L$	$C_D$	$C_m$
1	NACA4D_05	0.002750	0.003425	0.000968	0.999990	0.904181	0.999922
2	<b>NACA4D_10</b>	<b>0.002267</b>	<b>0.003147</b>	<b>0.000868</b>	<b>0.999994</b>	<b>0.914501</b>	<b>0.999940</b>
3	NACA4D_15	0.002013	0.003299	0.000912	0.999995	0.911276	0.999931

### Network Performance on Different Datasets

In addition to comparing dataset resolution, this project also examined how well the trained network could generalize beyond the specific airfoil family on which it was trained. The central question was whether a model trained exclusively on NACA 4-digit profiles could successfully predict aerodynamic coefficients for 5-digit airfoils, and vice-versa. The results presented in Table XV show that the network demonstrates strong generalization capability.

Table XV: Network Performance on Different Datasets [512, 256, 128, 3]

Case No.	Dataset		RMSE			$R^2$		
	Train	Test	$C_L$	$C_D$	$C_m$	$C_L$	$C_D$	$C_m$
1	NACA4D_10	NACA5D_10	0.061315	0.008127	0.030858	0.995584	0.471389	0.881492
2	NACA5D_10	NACA4D_10	0.026671	0.005698	0.011083	0.999092	0.722853	0.988516
3	<b>NACA4D_10 U NACA5D_10</b>	<b>NACA4D_10 U NACA5D_10</b>	<b>0.002138</b>	<b>0.003652</b>	<b>0.000947</b>	<b>0.999995</b>	<b>0.890248</b>	<b>0.999941</b>

Further inspection suggests that the artificial neural network does not memorize individual airfoil profiles, but instead learns the broader geometric design space represented by the coordinate distributions. Therefore, as long as a new airfoil lies within this learned geometric domain, the model is able to predict aerodynamic behavior with consistently high accuracy.

## V. Conclusion

This project presents an approach for predicting lift, drag, and moment coefficients using artificial neural networks (ANNs) trained on sparse, normalized two-dimensional airfoil coordinate data. The training datasets, consisting of NACA 4-digit and 5-digit airfoils under varying flight conditions, were generated using JavaFoil. Multiple network architectures were developed, trained, and systematically compared to determine the most effective configuration.

Following architecture selection, the network's performance was evaluated by altering the resolution of the geometric input data. The results demonstrated that the artificial neural network is capable of accurately capturing aerodynamic behavior even when provided with minimal coordinate information—unlike conventional CFD-based analysis, which typically requires detailed geometry and mesh refinement.

An additional finding was that the network learns the broader geometric design space rather than memorizing individual airfoil shapes. As a result, a network trained solely on NACA 4-digit profiles was still able to predict aerodynamic coefficients for NACA 5-digit geometries with acceptable accuracy, and the reverse condition showed similar behavior.

Overall, the study confirms that ANNs can rapidly and effectively learn aerodynamic characteristics using limited geometric input. This data-driven prediction capability offers a practical alternative for applications such as fully coupled aero-servo-elastic simulations, where fast execution and sufficient accuracy are essential—achieving results in a fraction of the time and computational cost required by traditional numerical simulation methods.

# **References**

<https://arxiv.org/pdf/2109.12149>

<https://ntrs.nasa.gov/api/citations/20020051082/downloads/20020051082.pdf>

<https://coxengines.ca/cox/www.mh-aerotools.de/airfoils/javafoil.htm>

<http://airfoiltools.com/>