encit 2020

ABCM Associação Brasileira de Engenharia e Ciências Mecânicas

18th Brazilian Congress of Thermal Sciences and Engineering
November 16–20, 2020 (Online)

ENC-2020-0395
NEURAL NETWORK ALGORITHM FOR PREDICTION OF AERODYNAMIC COEFFICIENTS OF A REDUCED SCALE ROCKET

Laura Pereira de Castro
Rodrigo Daher
Alexandre Zuquete Guarato
Federal University of Uberlândia, Av. João Naves de Ávila, 2121 - Santa Mônica, Uberlândia - MG, 38408-100
daher.rodrigo@ufu.br, laurapdec@ufu.br, azguarato@ufu.br

Abstract. *Computational Fluid Dynamics (CFD) simulations are extremely important in the fluid mechanics field, as they allow to obtain aerodynamic data without having the need to perform experimental simulations. On the other hand, machine learning, in a simplistic way, allows the computer to understand the behavioral connections between the inputs and outputs. The Aerospace Technology and Propulsion Team (EPTA) design and manufactures reduced scale rockets for competitions from Federal University of Uberlândia. This paper aims to present a neural network algorithm developed to predict the rockets aerodynamic data, so it would be possible to easily obtain the aerodynamic data and simulate the trajectory of the rocket without the need of experimental launches. In this paper CFD simulations were performed on Ansys © software and the neural network was developed thought a python algorithm. Aerodynamic data from the proposed algorithm have been compared to a similar rocket model using RASAero II © software, which is based on Barrowman's method to compute the aerodynamic coefficients. Results from the proposed neural network algorithm have a maximum 17.74 % deviation in relation to Ansys © and RASAero II ©, which is accurate enough to be used in the team.*

Keywords: *CFD, Aerodynamics, Machine Learning, Neural Networks*

1. INTRODUCTION

The Aerospace Technology and Propulsion Team (EPTA) is a group of Engineering students from Federal University of Uberlândia that designs and constructs reduced scale rockets. The first EPTA's rocket, a.k.a Vesper, is currently under development and it will achieve a 500 meters apogee.

In order to avoid problems with the project during the flight, which might have a high cost to correct, it's an interesting approach to use launch simulators. As the rocket has high construction costs launch simulations are a good approach to start the development. There are many launch simulator for rocket model such as RASAero II ©(Rogers and Cooper, 2016). However, they rely on Barrowman method (Barrowman, 1967), from calculating the aerodynamics coefficients. Although this method is fast but not as reliable as a CFD simulation.

Hence, a launch simulator algorithm is proposed in this paper for prediction of aerodynamic coefficients for any moment of the flight given the Mach number and the angle of attack. This paper presents CFD and prediction process of this algorithm.

A 3D CAD model of the rocket was designed with SolidWorks© software. Figure 1 presents the main dimensions of the the elliptical nose, body length, the inner diameter and outer diameter. Figure 2 presents the fins dimensions. All dimensions are in millimeters.

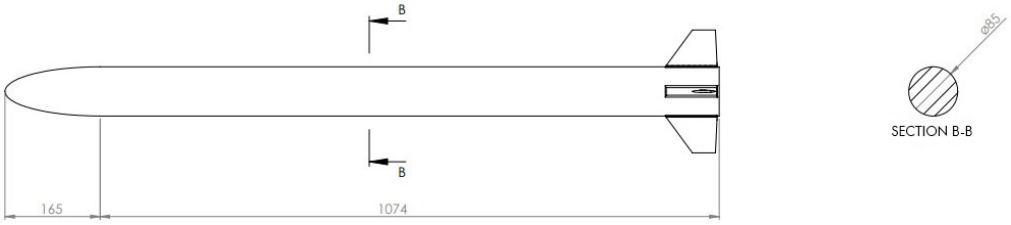

Figure 1: Rocket's dimensions.

To define the maximum possible velocity of the rocket was assessed using a simple ballistic equation ignoring the influence of the air. If there is no loss of energy for aerodynamic forces, the energy must be maintained, which means

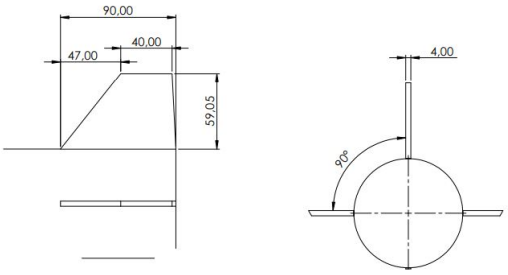

Figure 2: Fin's dimensions.

that the energy at highest point with only potential gravitational energy is the same as just after launch with the highest velocity (Eq. 1). The final rocket mass is smaller than the initial rocket mass since the fuel is burned during the flight. However, as a conservative measure, in order to estimate the maximum velocity which the rocket could ever reach, with a large safety margin, the mass is assumed to remain the same.

$$m_f g h = \frac{m_i V_{max}^2}{2} \tag{1}$$

where m_f is the final mass in Kg, m_i is the initial mass in Kg, V_{max} is the maximum velocity, h is the apogee and g is the gravitational acceleration

Therefore, it arrives at Eq. 2.

$$V_{max} = \sqrt{2gh} \tag{2}$$

Considering the gravity acceleration as $9.81\ m/s^2$ and $h = 500\ m$, the maximum obtained velocity is $99\ m/s$. Thus the rocket will be in subsonic and incompressible regime during all the flight.

In this paper the velocity was represented both in m/s and in the Mach number. In the aerodynamic simulations it was used velocity in m/s, once it was set in the program as so and to regard its accuracy. However, Mach number was used in the machine learning, since it is more common in papers and the rocketry literature.

2. AERODYNAMIC SIMULATIONS

CDF simulations were carried out using Ansys © in order to obtain the aerodynamic data of the rocket model. Thus, drag and lift coefficients were assessed. The steps to realize a CFD simulation in Ansys © are: geometry, mesh, setup and solution.

As said in the Introduction, the geometry of the rocket was designed using SolidWorks, firstly the CAD was imported to Ansys SpaceClaim © and after it was created the fluid domain. The fluid domain has to be 10x the dimension of the outer diameter around the rocket and 20x bigger than the same dimension behind the rocket to ensure the walls are not affecting the results of the simulation.

2.1 Mesh creation

The mesh is a fundamental part of a good simulation. It is always important to invest time creating the mesh to make sure it does not have any trouble and it is refined enough to the simulation. Also it is extremely important to make sure the mesh does not have too many elements, once the simulation requires computational process, a heavy mesh can escalate the time of the simulation and/or make impracticable to run in domestic computers. Usually 1 million elements requires almost 4GB RAM. With that information, the mesh was created making sure it would not overload 8GB of RAM, thus it could not have more than 2.5 million elements.

ICEM-CFD © software was used to create the mesh. This software has many important tools to define the global and local mesh parameters and it has an easy interface to work with. Except the boundary layer, all the mesh used were tetrahedral, while in external parts of the fluid domain, was defined a coarser mesh to make the simulation easier to run. However, the nose and the fins were more refined than the rest of the rocket, once they have rounded geometries it is harder to represent it as tetrahedrons. To solve this problem the tetrahedrons have to be smaller, resulting a more refined mesh.

Figure 3 shows the mesh from a cut plane, which shows the side of the fluid domain and the difference between the elements along the mesh. It is also possible to notice that behind the rocket there is a density box, an imaginary box of fluid which is more refined than the rest of the domain, the density box has the purpose of analysing more efficiently the behavior of the fluid after suffering the direct effects of rocket.

Figure 5 demonstrates the refinement along the rocket model and Figure 6 shows the refinement along the fins.

Lastly, a layer of prismatic mesh was created to ensure the accuracy on the surface of the rocket model. Whereas the boundary layer which is used to describe the contact zone between an incompressible fluid in motion relative to a solid, which the velocity changes from zero at the surface to the free stream value away from the surface. It is possible to visualize the prismatic layer in the mesh representing the boundary layer in the Figure 4.

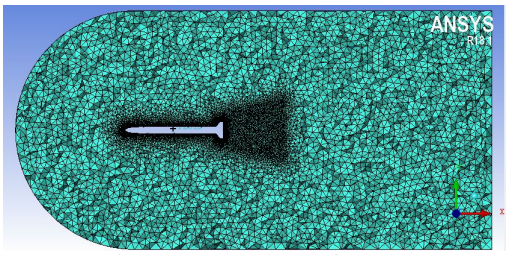

Figure 3: Cut plane of mesh.

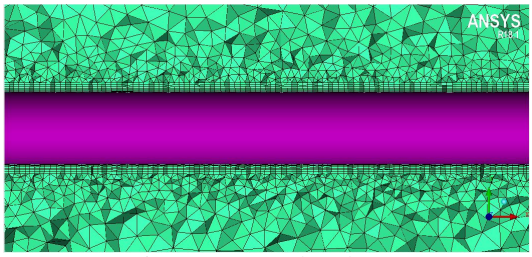

Figure 4: Boundary layer.

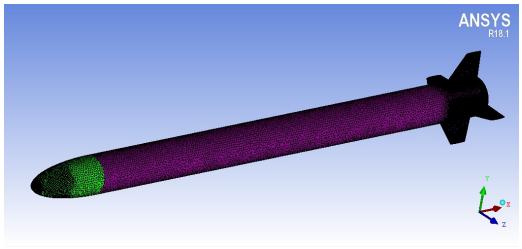

Figure 5: Rocket's mesh.

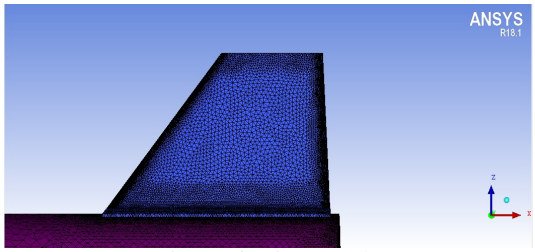

Figure 6: Fin's mesh.

2.2 Simulation Setup

The setup of the simulation determines the type of flow, if it is laminar or turbulent and the model of turbulence, if it is applicable. Also it determines the density and viscosity of the fluid, the velocity of the flow and the angle of attack in relation to the vertical. The aerodynamics coefficients, such as the drag coefficient and the lift coefficient, are set as the output.

The type of flow chosen was turbulent and the model of turbulence used was Spalart-Allmaras. The standard Spalart-Allmaras model (Spalart and Allmaras, 1992) is a one-equation model of turbulence that was designed specifically for aerospace applications which involves wall-bounded flows. This model was chosen due to its good results in the literature and its good computational performance, since it is a one-equation model the simulation runs much faster then other turbulent models. Also, the fluid was air in normal conditions, since the flow is incompressible, it has constant density of $1.225\ kg/m^3$ and constant viscosity of $1.7894 \times 10^{-05}\ Pa \cdot s$.

It was used 10 different velocities and 10 different angles of attack, resulting in 100 simulations. As said in the Introduction, the maximum velocity, ignoring the influence of the air, the rocket could reach would be $99\ m/s$, so the maximum velocity simulated was $91.1\ m/s$ once the rocket probably would not reach highers velocities. Rockets usually do not have high angles of attack, the objective is to keep $0°$ in relation to the vertical line, but due to the influence of horizontal wind gusts the rocket is susceptible to be with an angle of attack during the flight. The maximum angle of attack considered was $15°$, since highers angles could be dangerous to the rocket's flight.

The velocities and the angles used on the simulation are presented on Tab. 1

Table 1: Velocity and Angle of Attack Intervals.

Velocity [m/s]	20.0	27.9	35.8	43.7	51.6	59.5	67.4	75.3	83.2	91.1
Angle of attack [°]	0	1.5	3	4.5	6	7.5	9	10	12	15

Every simulation had 1000 iterations, and the convergence criteria was 1.0×10^{-05} to ensure the simulation would have all the iterations.

3. MACHINE LEARNING

An Artificial Neural Network (ANN) is an algorithm and one of the main fields of Deep Learning. The first neural network was the Perceptron (Rosenblatt, 1958) which was a single layer ANN represented at Fig. 7.

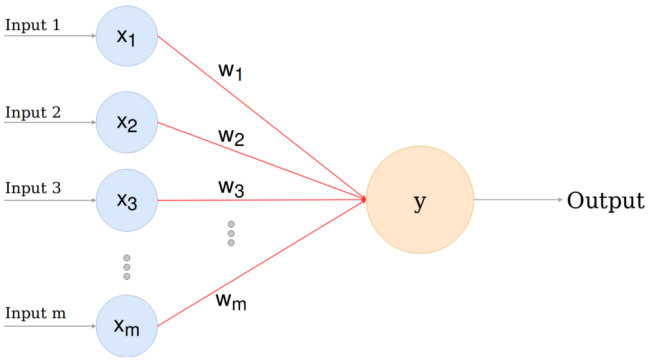

Figure 7: The Perceptron.

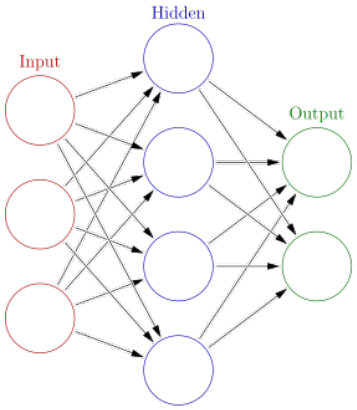

Figure 8: Neural Network Architecture.

It is easier to understand the concept of ANNs by understanding the Perceptron, the output is the result of the Eq. 3.

$$y = w_1 \times x_1 + w_2 \times x_2 + w_3 \times x_3 + \ldots + w_m \times x_m + bias \tag{3}$$

Where x_w is the value of the input k,w_k is the weight associated with the input k, $bias$ is a constant and y is the net input.

The weights and bias are initially random numbers that get updated during the training of the neural network in attempt to minimize the error associated. A neural network is a collection of neurons using the same logic as the Perceptron trying to find the best weights and biases so that given the input, the output will be closer to the one desired.

Finally, it's important to explain the activation functions. They are mathematical equations associated with each neuron of the ANN that determines its output based on the net input. In this paper the following activation functions were used: linear (Eq. 4), sigmoid (Eq. 5) and hyperbolic tangent (Eq. 6).

$$A = c \times x \tag{4}$$

$$A = \frac{1}{1 + e^{-x}} \tag{5}$$

$$A = tanh(x) \tag{6}$$

The neural network in this paper was designed using the Python library Pyrenn (Atabay, 2019). Considering that there are two inputs to the program, the Mach number and the angle of attack, the input layer has two neurons. The output layer also has two neurons since there are two outputs to the program, the drag coefficient and the lift coefficient.

When defining the architecture of the neural network, as stated in Thirumalainambi and Bardina (2003), a three layer neural network with the Levenberg-Marquardt Back Propagation training algorithm is sufficient to predict aerodynamic coefficients upon a given Mach number and angle of attack. The hidden layer neurons have been tested with the hyperbolic and sigmoid activation function while the output layer neurons have been tested with the linear and sigmoid activation function.

3.1 The Levenberg-Marquardt Back Propagation

The Levenberg-Marquardt method of optimization was firstly published by Levenberg (1944) and then perfected by Marquardt (1963). With the advance of neural networks, the back propagation method became one of the most popular due to its accuracy, however it has drawbacks like slow convergence speed and the local minima problem. To resolve this issue, the Levenberg-Marquardt method was inserted in the back propagation algorithm (Hagan and Menhaj, 1994), which resulted in a much more efficient training model.

4. RESULTS

In this work two types of results were obtained. Firstly, it was obtained the aerodynamic simulations results, which were the drag coefficient and the lift coefficient. Lastly, already with the aerodynamic data it was possible to run the machine learning code and train the data which gave the final results of this paper.

4.1 Aerodynamic Simulations

The drag coefficient margin considered reasonable were between 0.30 and 0.40. After finishing all the simulations, those which were considerate far from the expectation were simulated again.

Table 2: Drag Coefficient Table.

α \ V	20.0	27.9	35.8	43.7	51.6	59.5	67.4	75.3	83.2	91.1
0	0.33135	0.33465	0.35557	0.349448	0.33463	0.31348	0.33756	0.33475	0.33226	0.30356
1.5	0.34603	0.33463	0.35554	0.349455	0.34474	0.34083	0.33754	0.30782	0.33224	0.30352
3	0.34605	0.36382	0.35551	0.349437	0.34461	0.34072	0.33741	0.30770	0.33205	0.30337
4.5	0.34611	0.36383	0.35546	0.349287	0.34448	0.34052	0.33721	0.30749	0.33183	0.30316
6	0.34604	0.36370	0.35530	0.349093	0.34422	0.34028	0.33691	0.30724	0.33156	0.30288
7.5	0.34583	0.36345	0.35497	0.348741	0.34385	0.33983	0.33646	0.30676	0.33100	0.30238
9	0.34564	0.36314	0.35458	0.348293	0.31578	0.34012	0.33586	0.30619	0.33037	0.30178
10	0.34559	0.36291	0.35430	0.347948	0.31546	0.33890	0.33546	0.30581	0.32993	0.30134
12	0.34526	0.36248	0.35375	0.347329	0.31967	0.33806	0.33456	0.33154	0.32888	0.30031
15	0.34455	0.36355	0.35241	0.345748	0.34046	0.33624	0.33261	0.32949	0.32679	0.29838

Table 3: Lift Coefficient Table.

α \ V	20.0	27.9	35.8	43.7	51.6	59.5	67.4	75.3	83.2	91.1
0	-0.00155	0.00122	0.00237	0.00194	0.00884	0.00212	0.00206	0.00143	0.00239	0.00259
1.5	0.00919	0.00884	0.01018	0.01008	0.01009	0.01056	0.01042	0.00948	0.01043	0.00979
3	0.01617	0.01742	0.01811	0.01809	0.01813	0.01822	0.01824	0.01647	0.01796	0.01672
4.5	0.02357	0.02589	0.02646	0.02643	0.02654	0.02667	0.02669	0.02486	0.02675	0.02448
6	0.03134	0.03423	0.03496	0.03495	0.03531	0.03535	0.03541	0.03257	0.03543	0.03249
7.5	0.03959	0.04328	0.04381	0.04392	0.04439	0.04445	0.04448	0.04083	0.04443	0.04086
9	0.04857	0.06112	0.05351	0.05358	0.04956	0.05587	0.05431	0.04987	0.05425	0.05026
10	0.05499	0.06880	0.06030	0.06079	0.0558	0.06067	0.06096	0.05579	0.06079	0.05625
12	0.06895	0.07495	0.07534	0.07569	0.07087	0.07563	0.07565	0.07553	0.07553	0.06968
15	0.09299	0.11110	0.10066	0.10063	0.10058	0.10029	0.10019	0.09988	0.09963	0.09172

The drag and lift coefficients obtained from the simulations are shown in Table 2 and 3, respectively.
Figure 9 and Fig. 10 show a 3D plot of the obtained data, the drag coefficient and the lift coefficient, respectively.

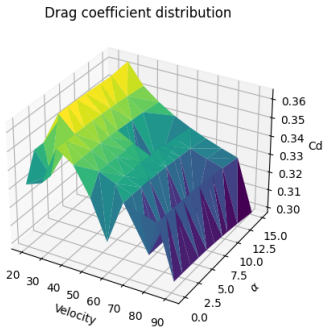

Figure 9: Drag Coefficient Distribution.

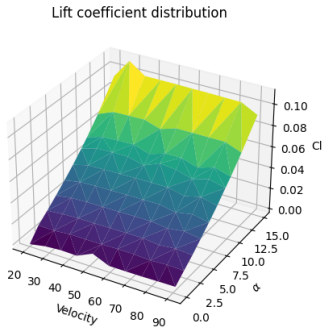

Figure 10: Lift Coefficient Distribution.

4.2 Machine Learning

The training set was defined as 80% of the total data set, the test set was defined as 15% and the validation set as 5%. Therefore there were 80 training data pairs, 25 testing data pairs and 5 validation data pairs.

Since there is no formula when it comes to defining the number of neurons in the hidden layer (N_h), such number was chosen based on maintaining a low error with a reasonable computational cost. At first, different activation functions were tested in the hidden and output layers all with $N_h = 30$ to evaluate which offers a better fitting. Figure 11 shows the result obtained with the sigmoid function in the hidden layer and the linear function in the output layer.

Figure 12 shows the result obtained with the sigmoid function in the hidden and output layer. Figure 13 shows the result obtained with the hyperbolic tangent function in the hidden layer and the linear function in the output layer. Figure 14

shows the result obtained with the hyperbolic tangent function in the hidden layer and the sigmoid function in the output layer.

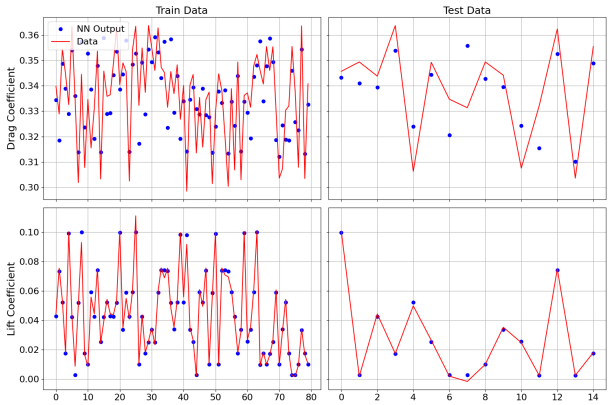

Figure 11: Sigmoid in the hidden layer and Linear in the output layer.

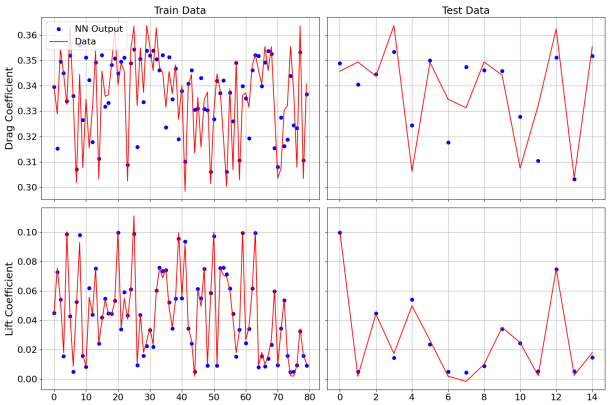

Figure 12: Sigmoid in the hidden layer and Sigmoid in the output layer.

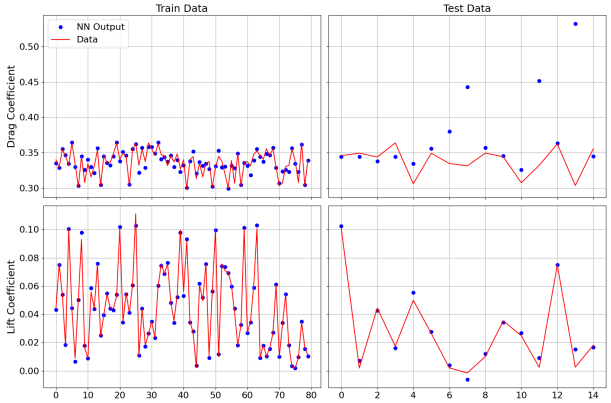

Figure 13: Hyperbolic Tangent in the hidden layer and Linear in the output layer.

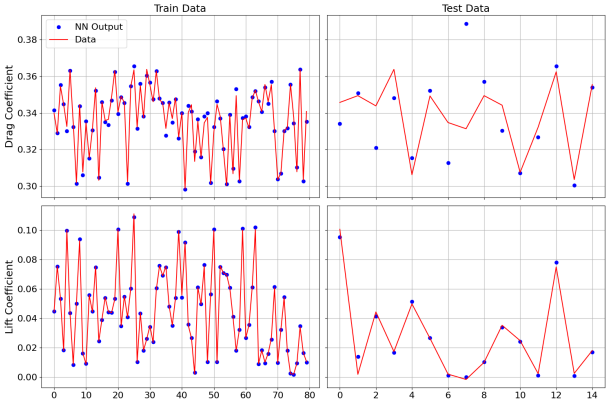

Figure 14: Hyperbolic Tangent in the hidden layer and Sigmoid in the output layer.

Comparing the results obtained, using the hyperbolic tangent in the hidden layer and the linear in the output layer has returned the smaller error, but it had some over fitting points in the test data. However, despite those points it had the most consistent behaviour and therefore this setup will be used to compare the difference of neurons in the hidden layer. Figures 15, 16 and 17 compare the results obtained with $N_h = 10$, $N_h = 30$ and $N_h = 50$ neurons, respectively.

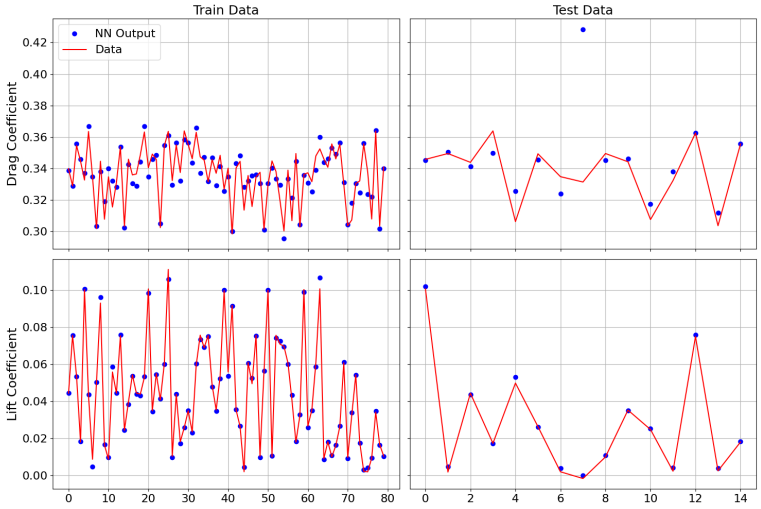

Figure 15: $N_h = 10$.

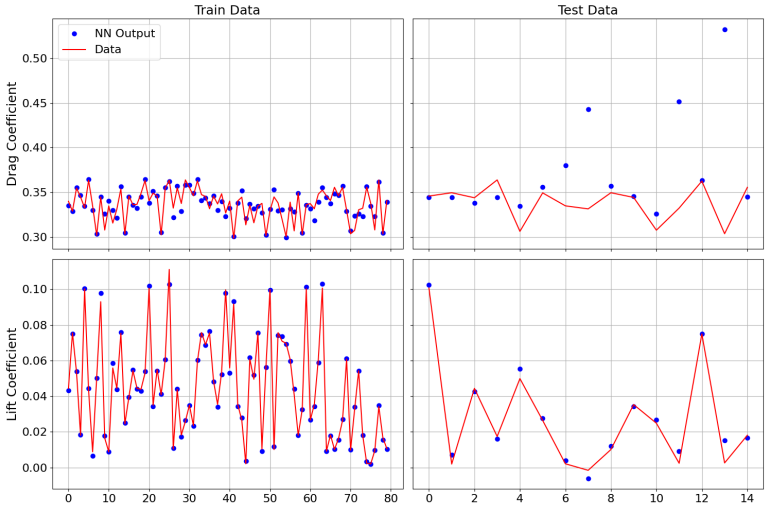

Figure 16: $N_h = 30$.

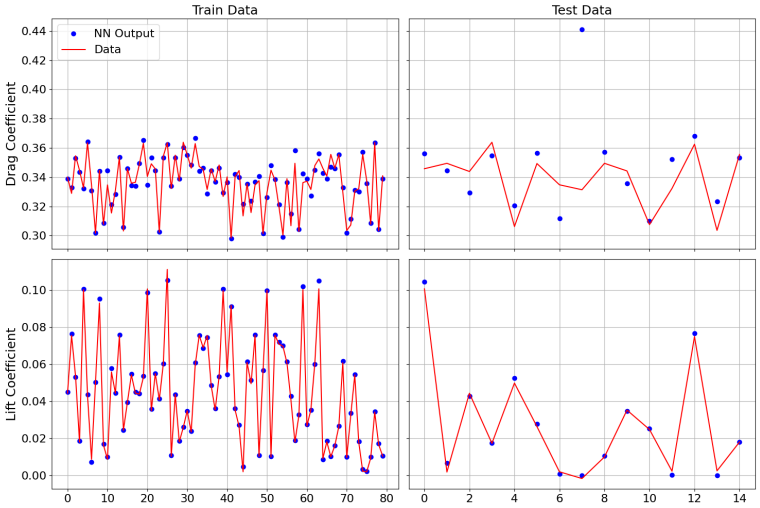

Figure 17: $N_h = 50$.

Finally, the ANN architecture with 50 neurons was defined as the final neuron network of this paper.

5. VALIDATION

The validation in this paper will be treated in two subsections: one for the aerodynamic data and another one for the neural network separately. To validate the drag coefficients and the lift coefficient at angle of attack 0°, it was compared with the data from the software RASAero II ©. For the lift coefficient with an angle of attack bigger than 0°, the neural network was used to calculate the aerodynamic coefficients to the angles of attack available at RASAero II ©. At last, the neural network was validated using the validation data set.

5.1 Aerodynamic Data Validation

It is important to highlight that this rocket model was fully designed by EPTA, which means it is not possible to find the same rocket in others papers to compare the aerodynamics coefficients.

To validate the data obtained the rocket model was assembled and simulated in the software RASAero II ©. The Table 4 compares the results of the drag coefficient.

Figure 18 presents the comparative graph of the data obtained in Ansys © simulations and RASAero II ©, at an angle of attack of 0°.

Table 5 shows the relative error for the aerodynamic coefficients between the CFD and RASAero II data ©. The maximum error presented in the drag coefficient is 17.28%, representing consistent results compared with RASAero II ©data. The lift coefficient error was presented by the absolute error once the RASAero II's ©results are zero, making it impossible to calculate the relative error. Hence, it is possible to verify that the errors of the lift coefficient are smaller

Table 4: Drag Coefficient Softwares Comparison.

V / Software	20.0	27.9	35.8	43.7	51.6	59.5	67.4	75.3	83.2	91.1
Ansys ⓒ	0.331	0.334	0.355	0.349	0.334	0.341	0.337	0.334	0.332	0.303
RASAero II ⓒ	0.374	0.376	0.377	0.375	0.374	0.373	0.371	0.370	0.369	0.367

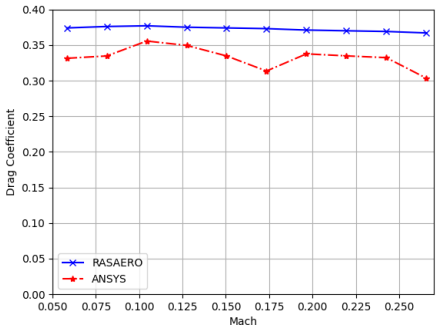

Figure 18: Drag Coefficient vs Velocity.

than 10^{-2}, demonstrating more accuracy than the drag coefficient.

Table 5: Relative and absolute error between CFD and RASAero II ⓒ.

Velocity [m/s]	20.0	27.9	35.8	43.7	51.6	59.5	67.4	75.3	83.2	91.1
Relative Error C_D	11.40%	11.00%	5.68%	6.81%	10.52%	15.96%	9.01%	9.53%	9.96%	17.28%
Absolute Error C_L	-0.0016	0.0012	0.0024	0.0019	0.0088	0.0021	0.0021	0.0021	0.0024	0.0026

There are some differences between the data obtained using Ansys ⓒ and RASAero II ⓒ. Those differences can be explained by many reasons. First of all, the velocities simulated with Ansys ⓒ and the data obtained by RASAero II ⓒ are not the same, so it was compared the closest velocity between those datas, which can provide a considerable error. Second, the methods to estimate the data are different, Ansys ⓒ realizes a CFD simulation, while RASAero II ⓒ uses the Barrowman's method. Also it is possible to say the turbulence model implemented and the airfoil in the fins which it is not possible to represent in RASAero II ⓒ, can also represent a significant discrepancy of the results.

It is important to discuss the results of the lift coefficient at 0° angle of attack. Theoretically the lift coefficient with no angle of attack is zero, which shows that the results obtained with the simulations represents an error, which can be due to a slightly asymmetry in the mesh.

5.2 Machine Learning Validation

The way to validate quantitative the machine learning was to separate 5%, therefore 5 data pairs, of the aerodynamic data and do not use it on the code, neither for test, nor for training. So, these 5 non-used results were compared with the result the code gave. This results can be seen in Fig. 19.

Table 6 presents the relative errors between the data predicted and the results obtained from CFD simulation. The maximum relative error obtained of 17.74% is which is more reliable than wind tunnel test, due to the influence of the rocket fixing rod (Faria *et al.*, 2019). Disregarding the over fitted data pairs the results are even more trustworthy with a maximum relative error of 7.00%.

Table 6: Relative error between the data predicted and CFD.

Data Pair	1	2	3	4	5
(Velocity [m/s] , α [°])	(83.2, 4.5)	(75.3,10.0)	(20.0,6.0)	(59.5,4.5)	(27.9,0.0)
C_D	1.05%	6.20%	0.66%	0.39%	17.60%
C_L	0.42%	7.00%	0.58%	4.59%	17.74%

Most of the predictions were correct although the last data pair had a dysfunctional prediction, it could probably be associated with the over fitting and disregarded for use. Nonetheless, the software can predict the data within an interval more reliable than the Barrowman's method, since it is based on CFD and much faster than a CFD simulation.

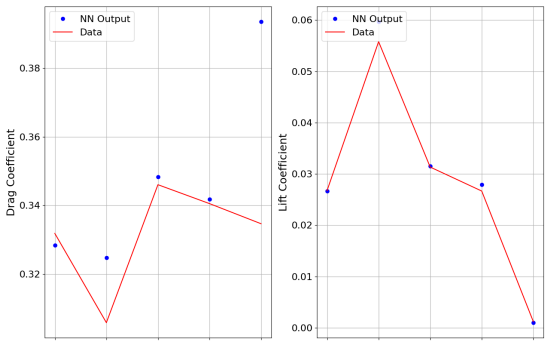

Figure 19: Validation Data Set.

In order to validate the lift coefficient at higher angles of attack, the neural network was used to calculate the aerodynamic coefficients to the angles of attack available at RASAero II ©. As shown in Fig. 20, the software RASAero II © has predicted high lift coefficients sometimes bigger than the drag coefficient, providing a relation between $C_D \times C_L$ which are not consistent with the literature (Natarajan, 2018). Thus the validation of the lift coefficient could not be made quantitatively, although qualitatively it can be easily seen that the data is showing the expected behaviour in terms of linearity related to the Mach number and angle of attack.

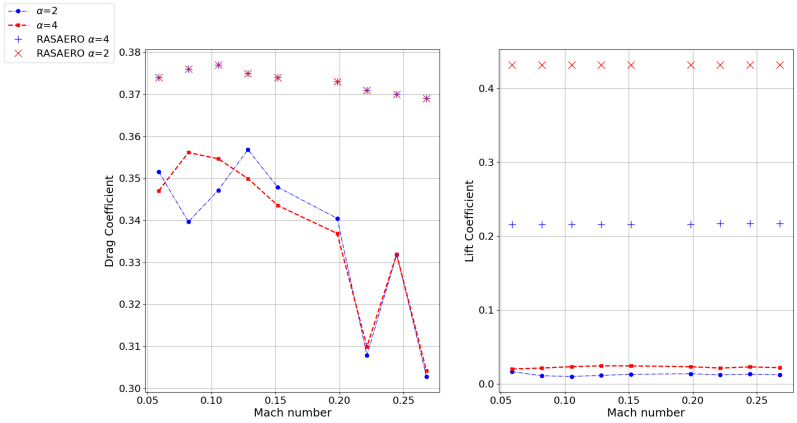

Figure 20: Validation of lift coefficient with RASAero II ©.

6. CONCLUSIONS

The main objective of this paper is to collect aerodynamic data by CFD simulations and develop a functional neural networks algorithm, so it could learn the behavior of the aerodynamics coefficients based on the Mach number and angle of attack. With the proposed algorithm aerodynamic coefficients can be predicted in other conditions.

The results of the ANN had an expected behaviour, especially those for the lift coefficient. However, the drag coefficient has not been so precise as expected, once the data collected by Ansys © showed medians results.

Regardless, the results were consistent and accurate, with a maximum error of 17.28% for the aerodynamic data and 17.74% for the ANN results, and they are suitable for the applications of the team. No further CFD simulations were necessary. Authors intend to improve drag coefficients estimations and compare the results of this work with experimental data.

7. ACKNOWLEDGEMENTS

The authors would like to express their gratitude to the Aerospace Technology and Propulsion Team (EPTA).

8. REFERENCES

Atabay, D., 2019. "Pyrenn: A recurrent neural network toolbox for python and matlab". `https://github.com/yabata/pyrenn`. doi:10.5281/zenodo.45022. Version 0.1.

Barrowman, J.S., 1967. *The practical calculation of the aerodynamic characteristics of slender finned vehicles*. Master's thesis, The Catholic University of America, Washington, D. C.

Faria, V.G.F.L.R., Bezas, V.C.S., Ribeiro, F.J.O. and Almeida, O., 2019. "Computational and experimental analyses of the airflow over a rocket model". In *Proceedings of the 25nd International Congress of Mechanical Engineering - COBEM 2019*. Uberlândia, Brazil.

Hagan, M.T. and Menhaj, M.B., 1994. "Training feedforward networks with the marquardt algorithm". *IEEE Transactions on Neural Networks*, Vol. 5, No. 6, pp. 989–993.

Levenberg, K., 1944. "A method for the solution of certain non-linear problems in least squares". *Quarterly of applied mathematics*, Vol. 2, No. 2, pp. 164–168.

Marquardt, D.W., 1963. "An algorithm for least-squares estimation of nonlinear parameters". *Journal of the society for Industrial and Applied Mathematics*, Vol. 11, No. 2, pp. 431–441.

Natarajan, G.S., 2018. *Design Of a personal aerial vehicle (PAV)- Roadable Aircraft (Flying Car)-"DROGON V-1"*. Ph.D. thesis. doi:10.13140/RG.2.2.10470.68163.

Rogers, C.E. and Cooper, D., 2016. "Rasaero ii: Rocket aerodynamic analysis and flight simulation software". *Rogers Aeroscience*.

Rosenblatt, F., 1958. "The perceptron: a probabilistic model for information storage and organization in the brain." *Psychological review*, Vol. 65, No. 6, p. 386.

Spalart, P.R. and Allmaras, S.R., 1992. "A one-equation turbulence model for aerodynamic flows". *30th Aerospace Sciences Meeting and Exhibit*.

Thirumalainambi, R. and Bardina, J., 2003. "Training data requirement for a neural network to predict aerodynamic coefficients". In *Independent Component Analyses, Wavelets, and Neural Networks*. International Society for Optics and Photonics, Vol. 5102, pp. 92–103.

9. RESPONSIBILITY NOTICE

The authors are solely responsible for the printed material included in this paper.