Avoid aeroelasticity instabilities with a morphing airfoil using neural networks

(Final report)

Raul Carreira Rufato* and Dr Joseph Morlier†
*Institut Supérieur de l'Aéronautique et de l'Espace (ISAE-SUPAERO), Université de Toulouse, 31055 Toulouse, FRANCE
Email: raul.carreira-rufato@student.isae-supaero.fr
†Institut Supérieur de l'Aéronautique et de l'Espace (ISAE-SUPAERO), Université de Toulouse, 31055 Toulouse, FRANCE
Email: joseph.morlier@isae-supaero.fr

***Abstract*—Avoiding an aircraft instability condition is extremely important, especially for flight control, and morphing airfoils can be used to avoid these situations. This work proposes the determination of a morphing airfoil, using a machine learning approach, for a given unstable condition. Instead of using full airfoil's coordinates, Bezier – PARSEC 3434 parameters have been used to describe them, were some of the have been determined using a Genetic Algorithm. The airfoil's Cg position, c_l, c_d and c_m distributions for some angles of attack have been put into a neural network for learning and then to be able to estimate the BP3434 parameters. Finally, the neural network was coupled to the 2D aeroelastic model of a wing.**

List of Symbols

c_l	lift coefficient
c_d	drag coefficient
c_m	moment coefficient
Cg	center of gravity
EA	elastic axis
ODE	ordinary differential equation
MSE	mean square error
ANN	artificial neural networks

I. Context

Historically, the study of airfoils has always been extremely important in the aeronautical industry, since they can be considered as the heart of the airplane [15]. Airfoils are also essential in describing the aeroelastic characteristics of a wing, so studying and improving their performance is highly necessary. In this context, morphing airfoils are being studied, due to their ability to adapt to a given flight requirement, as this technology is aimed at very efficient aerodynamic and structural designs during flight, contributing to the high performance of aircraft [21]. The impact of this new technology, as well as simple modelling of this type of airfoil were better described in [3]. They are similar to the birds' ability to adapt their wings to certain flight conditions, thus improving their performance.

In an aircraft, several situations that occur during flight can alter the global center of gravity and even that of the wing, such as jettisoning and fuel consumption, which can instantly bring the aircraft into an unstable condition. Therefore, using airfoils of variable geometry allows the optimization of their shape, hence returning to a stable position. Due to the high non-linearity of this process, it is interesting to use artificial neural networks which are viable computational models aimed at a wide variety of problems, including optimization, nonlinear system modelling and control [10]. This seeks to represent the way the human brain works, being provided with neurons and connections between them. This modelling is one of the most modern ways to optimize highly complex systems and obtain acceptable results. This work links the study of the aeroelastic characteristics of morphing airfoils with the neural network tool.

II. Problem statement

Despite being a simplistic model, with a reduced number of degrees of freedom and in 2D, a few problems can occur, as in [11]:

- For the construction of the database there is the need to carry out an optimization by differential evolution for each airfoil.
- To guarantee the training of a neural network with high accuracy and low loss function, a high number of airfoils and their respective parametrization is necessary, which further increases the time of the process.
- Difficulty in coupling the neural network model with the aeroelastic model.

Despite the problems that can occur, this research aims to develop a simplified model, although being very representative for this case study.

III. State of Art

Determining an optimal geometry for the morphing airfoil consists of a highly complex optimization problem because, known airfoils are described by a vector of 50–80 length, as listed in the UIUC Airfoil Coordinates Database [18]. However, using a form of parameterization through Bezier-Parsec curves as described in [17], it is possible to describe the geometric characteristics of an airfoil through a reduced number of parameters. Bezier's camber-thickness formulation is more directly related to flow than upper curve-lower curve formulation for PARSEC, while PARSEC parameters are more aerodynamically oriented than Bezier parameters. The BP parameterization uses the PARSEC variables as parameters, which in turn define four separate Bezier curves. These curves describe the leading and trailing portions of the camber line, and the leading and trailing portions of the thickness distributions [6].

Using the BP3434 parameterization it is possible to train a neural network as in [11], for a desired input vector. In this work, we chose to use the same idea to train a neural network using the Keras package in Python. Additionally, the coupling will be made with an aeroelastic simulation in which an airfoil will have undergone a sudden change in its Cg, taking it to an unstable condition. For the modeling of this condition, a model like the one in [13] will be made. The neural network will then be able to determine a new airfoil (through BP3434 curves) to return the Cg to a stable condition, maintaining the same aerodynamic characteristics as the current profile.

To conclude, the construction of the database to train the neural network will be done using the Matlab software and the Xfoil code. Then the neural network and the aeroelastic code will be coupled in Python language.

IV. Background

A. Artificial neural networks

Artificial neural networks (ANN) are computational models designed to simulate the way the human brain analyses and processes information. Such systems learn to perform tasks by considering examples and they are formed by a set of processing units called neurons. For every neuron there is a state of activation y_j, which is equivalent to the output of the unit. The network consists of connections between these neurons, each connection providing an output which is then an input to another neuron. For each of these connections a weight w_{ij}, that represents its relative importance as well as the effect of the neuron i on neuron j, is assigned [23]. Each neuron also has an external input (bias) b_j that shifts the function up or down. Then, the product input-weight and the bias are added up and passed through a node's so-called activation function, to determine whether and to what extent that signal should progress further through the network to affect the ultimate outcome, in other words, an act of classification. Most frequently the form of the propagation rule is given as [11].

$$y_j(t) = \sum_{i=1}^{N} w_{ij}(t)x_j(t) + b_j(t) \tag{1}$$

Next, an activation function $\phi(\bullet)$ is given and it then determines the new level of activation based on the effective input $x_j(t)$ (Figure 1):

Thus, each neuron receives an input from its neighbours or external sources and uses this to compute an output signal which is propagated to another neuron. The connections between them build the ANN as can be seen in Figure 2. Additionally, each column section of these neurons is called a layer. The second task is to adjust the weights so a learning method for information gathering about the error is made by the ANN to adjust the value of the individual weights w_{ij}.

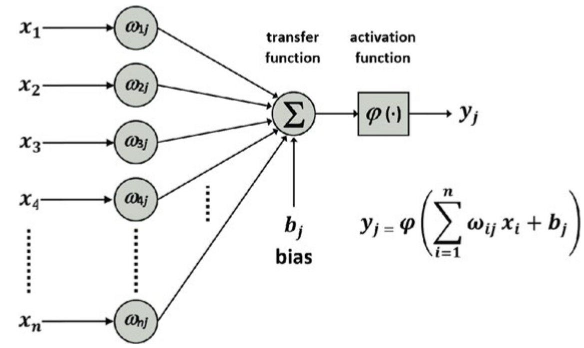

Fig. 1. Structure of one neuron.

This is seen as a parallel system given that many neurons can carry out their computations at the same time.

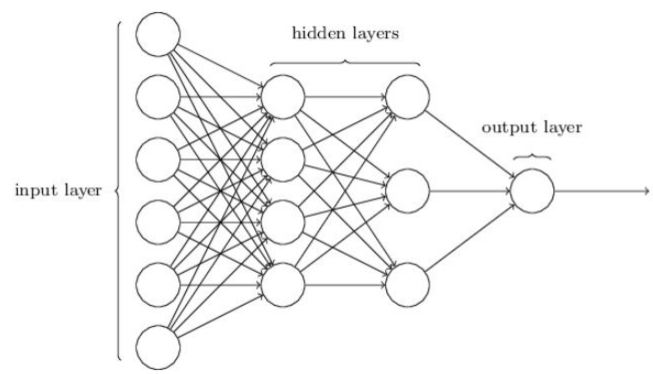

Fig. 2. Structure of an ANN.

Moreover, it is important to select the correct activation function as it performs the non-linear transformation in the input data, making it possible to learn and perform more complex tasks. This function enables later propagation as long as the gradients are provided together with the error to update the weights and bias, therefore, without the differential non-linear function this would not be possible. The aim of this work is to predict the BP3434 parameters given the aerodynamic coefficients and the Cg position as inputs, resulting in a regression task. There are several types of activation functions, such as Linear, Sigmoide, Tanh, ReLU, Leaky ReLU, Softmax, in which Leaky ReLU was selected for this work. However, it is first necessary to define the ReLU function, which is the rectified linear unit. The ReLU function is defined as:

$$f(x) = max(0, x) \tag{2}$$

It is the most used activation function when designing neural networks. The ReLU function is non-linear, which means that it can easily copy the errors back and have multiple layers of neurons activated by the ReLU function. The main advantage of using the ReLU function over other activation functions is that it does not activate all neurons at the same time. However, in the ReLU function, the gradient is 0 to $x < 0$, which causes neurons to die from activations in that region. Hence the use of Leaky ReLU, which helps to solve this problem. The Leaky ReLU function is defined as:

$$\begin{aligned} f(x) = ax,\ x < 0 \\ f(x) = x,\ x \geq 0 \end{aligned} \tag{3}$$

so the ReLU graph ends up having a small slope and avoiding null values for the gradient.

B. Genetic algorithms

The genetic algorithm is a method for solving both constrained and unconstrained optimization problems that is based on natural selection [11]. This optimization algorithm is interesting because there is no need to know the function or even its derivative to be able to do the desired optimization. Its convergence is not proved mathematically although it always converges to an optimal value.

At each step, the genetic algorithm selects the best individuals from the current population and combines their characteristics randomly to produce the new generation of candidates. The first population is generated randomly but after several iterations, the population evolves to an optimal solution. The algorithm uses three main rules to evaluate and generate its population: the selection rule chooses the best individuals to propagate their characteristics to the next generation; the crossover rule combines the characteristics of these individuals to generate the children and the mutation rule applies random changes from parents to children.

In this work, the genetic algorithm was used to optimize some parameters of the BP3434 curve from the airfoil curve, consequently producing the database. This optimization was used because the function of this transformation was not known. It was performed according to [11].

C. Bezier curves

A Bezier curve of degree n is defined by $n+1$ points of a polygon. These vertices are the control points which are physical points in the plane. To describe the airfoil four Bezier curves were used, two for the camber line and two for the thickness distribution. The general expression for one curve with degree n is:

$$P(u) = \sum_{i=0}^{n} P_i \frac{n!}{i!(n-i)!} u^i (1-u)^{n-i} \tag{4}$$

Where P_i represents a control point. The parameter u goes from 0 to 1 with 0 at the P_0 control point and 1 at the P_n control point.

This work will use a third degree curve and a fourth degree curve that will be later explained. A third order Bezier curve is given by the following equations:

$$\begin{aligned} x(u) &= x_0(1-u)^3 + 3x_1u(1-u)^2 + 3x_2u^2(1-u) \\ &\quad + x_3u^3 \\ y(u) &= y_0(1-u)^3 + 3y_1u(1-u)^2 + 3y_2u^2(1-u) \\ &\quad + y_3u^3 \end{aligned} \tag{5}$$

And a fourth order Bezier curve is given by the following equations:

$$\begin{aligned} x(u) &= x_0(1-u)^4 + 4x_1u(1-u)^3 + 6x_2u^2(1-u)^2 \\ &\quad + 4x_3u^3(1-u) + x_4u^4 \\ y(u) &= y_0(1-u)^4 + 4y_1u(1-u)^3 + 6y_2u^2(1-u)^2 \\ &\quad + 4y_3u^3(1-u) + y_4u^4 \end{aligned} \tag{6}$$

An airfoil can be described by the Bezier curves, however the problem with this parametrization is that it does not establish a proper relationship with the airfoil's aerodynamic parameters, it is merely geometric related.

D. PARSEC parameters

The PARSEC method is a very common and highly effective method of airfoil parameterization. It uses eleven basic parameters to completely define the aerofoil shape which are the leading edge radius, upper crest location, lower crest location, upper and lower curvature, trailing edge coordinate and direction, trailing edge wedge angle and thickness [17]. This method incorporates the parameters that have a physical relevance to the airfoil shape.

Unfortunately, PARSEC does not provide sufficient control over the trailing edge shape where important flow phenomenon can take place because it fits a smooth curve between the maximum thickness point and the trailing edge, making changes difficult at the trailing edge [2].

E. Bezier–PARSEC parameterization

Combining both methods described before, the Bezier-PARSEC can parametrize a given airfoil with parameters that are related to its aerodynamic and geometrical characteristics. The detailed development of this method is given by [20]. As said in [17], Oyama et al. [14] showed that Bezier-PARSEC parametrization increased the robustness and convergence speed for aerodynamic optimization using genetic algorithms. It has a lot of advantages and in this work it will be used to generate the airfoil shape, building the database to train the neural network architecture.

The Berzier-PARSEC parametrization uses the PARSEC variables as parameters, which in turn define four separated Bezier curves, two curves to describe the leading edge of the thickness curve & camber curve and two to describe the trailing edge for both of them. In this work a BP3434 parametrization will be used. As in the name, both leading edge curves have polynomials of third degree while both the trailing edge curves have the polynomials of fourth degree. In Figure 3 these curves can be seen as well as the Bezier control points and the ten PARSEC parameters. The parametrization assumes a unitary chord for the airfoil.

The parameters in the Equations 5 and 6 can be determined as follows:

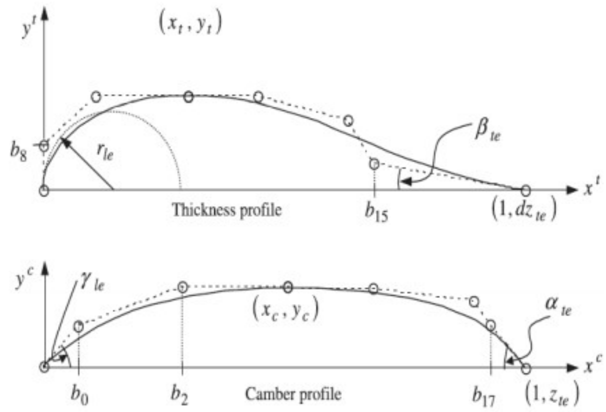

Fig. 3. BP 3434 airfoil geometry and Bezier control points defined by ten aerodynamic and five Bezier parameters [17]

1) Leading edge thickness curve: The control points can be described as:

$$
\begin{aligned}
(x_0, y_0) &= (0, 0);\\
(x_1, y_1) &= (0, b_8);\\
(x_2, y_2) &= (\frac{-3b_8^2}{2r_{le}}, y_t);\\
(x_3, y_3) &= (x_t, yt);
\end{aligned}
$$

In which b_8 is between $0 < b_8 < min(y_t, \sqrt{\frac{-2r_{le}x_t}{3}})$

2) Trailing edge thickness curve: The control points can be described as:

$$
\begin{aligned}
(x_0, y_0) &= (x_t, y_t);\\
(x_1, y_1) &= (\frac{7x_t + 9b_8^2}{8r_{le}}, y_t);\\
(x_2, y_2) &= (3x_t + \frac{15b_8^2}{4r_{le}}, \frac{y_t + b_8}{2});\\
(x_3, y_3) &= (b_{15}, dz_{te} + (1 - b_15)tan(\beta_{te}));\\
(x_4, y_4) &= (1, dz_{te});
\end{aligned}
$$

3) Leading edge camber curve: The control points can be described as:

$$
\begin{aligned}
(x_0, y_0) &= (0, 0);\\
(x_1, y_1) &= (b_0, b_0tan(\lambda_{le}));\\
(x_2, y_2) &= (b_2, y_c);\\
(x_3, y_3) &= (x_c, y_c);
\end{aligned}
$$

4) Trailing edge camber curve: The control points can be described as:

$$
\begin{aligned}
(x_0, y_0) &= (x_c, y_c);\\
(x_1, y_1) &= (\frac{3x_c - y_c cot(\lambda_{le})}{2}, y_c);\\
(x_2, y_2) &= (\frac{-8y_c cot(\lambda_{le}) + 13x_c}{6}, \frac{5y_c}{6});\\
(x_3, y_3) &= (b_{17}, z_{te} - (1 - b_{17})tan(\alpha_{te}));\\
(x_4, y_4) &= (1, z_{te});
\end{aligned}
$$

F. Aeroelastic Model

The aeroelastic model chosen in this work is a simplified model of an airfoil with two degrees of freedom: Translation and rotation. A schematic representation of the system can be seen in Figure 4.

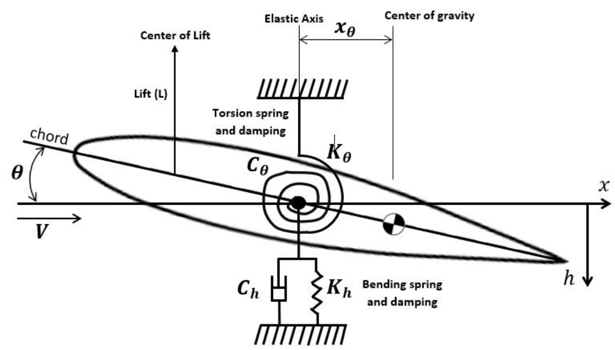

Fig. 4. Modeled system. Font: Author

The airfoil has a damping and stiffness system in both degrees of freedom, therefore it is possible to model the system as in [8]. Knowing that x is measured along the chord and it is not a generalized coordinate, it cannot undergo virtual change. The generalized coordinates can be described as:

$$\{q_1 = h,\ q_2 = \theta\} \tag{7}$$

And the displacement of any point in the airfoil is

$$\overrightarrow{r} = u\,\overrightarrow{i} + z\,\overrightarrow{k} \tag{8}$$

Where u is the horizontal displacement component and z is the vertical displacement component. From the geometric characteristics it can be said:

$$\theta \ll 1 \rightarrow \begin{cases} u = x[cos\theta - 1] \approx 0 \\ z = -h - xsin\theta \approx -h - x\theta \end{cases} \tag{9}$$

Hence, the kinetic energy is calculated as:

$$
\begin{aligned}
T &= \frac{1}{2}\int[(\frac{dz}{dt})^2 + (\frac{du}{dt})^2]\rho dx \\
&\simeq \frac{1}{2}\int(\frac{dz}{dt})^2\rho dx \\
&= \frac{1}{2}\int(-\dot{h} - \dot{\theta}x)^2\rho dx \\
&= \frac{1}{2}\dot{h}^2\int\rho dx + \dot{h}\dot{\theta}\int x\rho dx + \frac{1}{2}\dot{\theta}^2\int x^2\rho dx \\
&= \frac{1}{2}\dot{h}^2 m + \dot{h}\dot{\theta}x_\theta m + \frac{1}{2}\dot{\theta}^2 I_\theta
\end{aligned} \tag{10}
$$

Where $m = \int \rho dx$ is the total mass; $I_\theta = \int \rho x^2 dx$ is the moment of inertia and ρ is the mass per unit chord length.

The potential energy is:

$$U = \frac{1}{2}K_h h^2 + \frac{1}{2}K_\theta \theta^2 \tag{11}$$

where K_h and K_θ are the spring stiffness. Then, calculating the generalized forces and using the Lagrange's equations, with the right algebraic arrangement, it is possible to obtain the equation of motion that describes the model (Equation 12):

$$
\begin{aligned}
&\begin{bmatrix} m & mx_\theta \\ mx_\theta & I_\theta \end{bmatrix} \times \begin{bmatrix} \ddot{h} \\ \ddot{\theta} \end{bmatrix} + \begin{bmatrix} C_h & 0 \\ 0 & C_\theta \end{bmatrix} \times \begin{bmatrix} \dot{h} \\ \dot{\theta} \end{bmatrix} + \\
&\qquad \begin{bmatrix} K_h & 0 \\ 0 & K_\theta \end{bmatrix} \times \begin{bmatrix} h \\ \theta \end{bmatrix} = \begin{bmatrix} F \\ M \end{bmatrix}
\end{aligned} \tag{12}
$$

In which m represents the mass of the airfoil; x_θ represents the distance between the Cg (center of gravity) and the elastic axis and $I_\theta = I_0 + mx_\theta^2$ is the moment of inertia displaced from the Cg; $h(t)$ is the degree of freedom plunge and $\theta(t)$ is the degree of freedom in torsion, both measured from the static equilibrium position.

Considering this a dynamic aeroelasticity problem, it was used an aerodynamic model to describe the flow. Thus, the panel method was used to determine the aerodynamic loads during the time simulation. Because aerodynamic and structural models have different requirements, it is necessary to use different approaches for the discretization of the modelled geometry, consequently , the aerodynamic and structural meshes are different, containing an interface which provides the communication between them. The solution for the interface used, provided by ALTRAN, uses a Radial-Basis-Function (RBF), a method for spatial interpolation in 2D, which transfers the loads from the structural mesh to the aerodynamic mesh.

V. Methodology

The diagram that represents the order of development of the work can be seen in Figure 5.

The work began with the construction of the airfoil database and its respective parameters, to train the neural network. In parallel to this process, the aeroelastic code was developed implementing the described system, as well as the possibility of changing the center of gravity to an unstable position. Then, after analysing the validity of the results for both models, it was possible to train the neural network and couple it with the fluid-structure interface. The validations of each step are presented in the following sections.

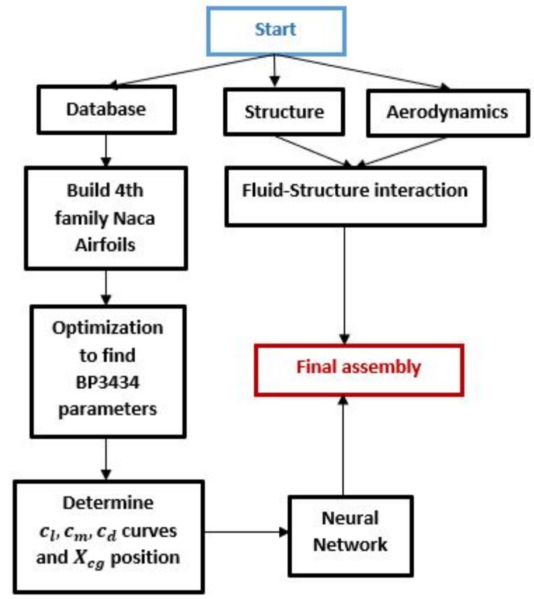

Fig. 5. Project main scheme

TABLE I
OPTIMIZATION CHARACTERISTICS

Objective Function	Square error (real and fitted)
Bounds	Constrained
Type	Without derivate
Number of members of the population	120
Desirable value for objective function	1e-6
Number of variables to optimize	8
Crossover probability	0.8
Maximum number of iterations (generations)	50

A. Construction and validation of the database

The research development started by building the database, which is important for training the neural network.

First, the x and y coordinates were determined for the airfoil points of the 4^{th} NACA Family, varying: maximum thickness, maximum camber and location of maximum camber, where a thousand different airfoils of the same family were built. An optimization was then performed to determine the parameters of Bezier-Parsec 3434 that would be able to describe them. However, in order to reduce the number of optimization parameters, it was possible to determine seven of them analytically (d_{zte}, z_{te}, r_{le}, x_c, y_c, x_t and y_t), because they are geometric characteristics of each profile, such as the maximum thickness and maximum camber. The remaining eight parameters have been optimized, using the differential evolution (DE) algorithm of Rainer Storn [19], in MATLAB through parallel simulation processing. The characteristics of the optimization can be seen in Table 1.

Finally, it was possible to determine the control points of the airfoil and estimate its curve using the relationships between camber and thickness lines. The results of this approach can be seen in Figures 6 and 7, for the NACA 2412 airfoil.

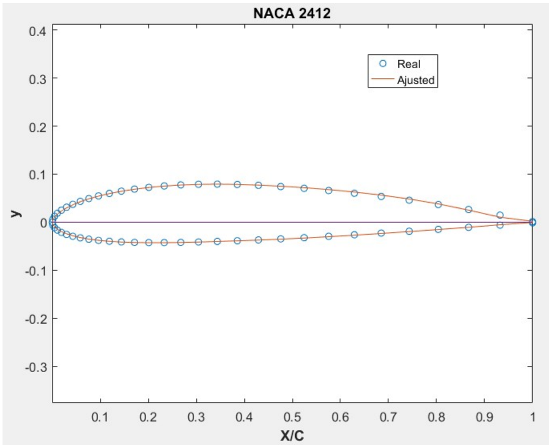

Fig. 6. BP 3434 parametrization: control points. Font: Author

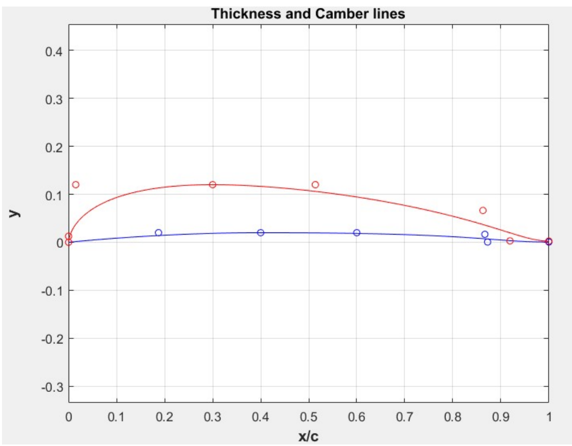

Fig. 7. BP 3434 airfoil geometry aproximation using Bezier-Parsec 3434 parameters. Font: Author

It is possible to see through Figure 6 that the curve adjustment was satisfactory given the number of points that describes the airfoil. Figure 7 shows the control points to describe the thickness and camber lines, determined by the Bezier-Parsec parameters.

The curves of c_l, c_m, c_d (for angles of attack from 1 to 11) and the position of X_{cg} for each airfoil were determined using the code Xfoil [9]. In possession of the aerodynamic curves and the Bezier-Parsec parameters of each profile, the necessary database for the training of the neural network was built.

B. Neural network training

Using the built database, the problem to be solved by the neural network consists of predicting the 15 parameters for a Bezier-Parsec 3434 parametrization, given c_l, c_m and c_d curves and the X_{cg} of an unknown airfoil, therefore it is a regression problem.

TABLE II
DEEP FEED FORWARD (DFF)

Layers	3
Hidden layers	LeakyReLU
Architecture	500-100-50-15
Output layer	No activation function
Optimizer	Adam (learning rate=0.001)
Loss	MSE
Data division	Training 80% ; Test 20%

For the construction of this neural network, the Keras library was used in python [5], generating a Deep Feed Forward (DFF) type architecture. The DFF is basically an artificial neural network in which connections between the nodes do not form a cycle [22]. This is the simplest type of neural network, as the information moves unidirectionally, from the input layer - through the hidden layers, if any - to the output layer, with no loops or cycles, where all neurons in one layer are fully connected with those in the next layer. The information on the neural network implemented in this work is described in Table 2.

Three hidden layers with 500,100 and 50 neurons respectively were used, all of them with the LeakyReLU activation function explained previously in Section IV. The Output layer must not have any activation function, as it is a regression task, in which the values must be predicted and not classified. The Adam optimizer with a low learning rate [7] was used to train the network and determine the optimal weights for each connection. The loss function chosen was the Mean Square Error (MSE) function, described by Equation 13:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \widetilde{y}_i)^2 \tag{13}$$

The database was used for two different reasons, being 80% for training the neural network and 20% for testing. The training data was not normalized to mean 0 and standard deviation 1 due to the fact that they already have an average close to zero and a very small standard deviation.

Using a database of 1000 4^{th} NACA family airfoils and their respective parameters, it was possible to train the neural network and test it. Additionally, its quality was analyzed using the K-fold cross-validation method, which is a resampling procedure used to evaluate machine learning models on a limited data sample. The procedure has a single parameter called k that refers to the number of groups that a given data sample is to be split into. As such, the procedure is often called k-fold cross-validation [4]. The validation process also described at [4] consists of:

1) Shuffle the dataset randomly.
2) Split the dataset into k groups.
3) For each unique group:
    a) Take the group as a hold out or test data set;
    b) Take the remaining groups as a training data set;
    c) Fit a model on the training set and evaluate it on the test set;

4) Retain the evaluation score and discard the model.
5) Summarize the skill of the model using the sample of model evaluation scores

Using $k = 10$ number of splits in the dataset, an average result equal to -0.02421460 and standard deviation equal to 0.00417892, representing a satisfactory result for this data set. Therefore, by training the neural network, it was possible to test it. The results can be seen in Figures 8, 9 and 10.

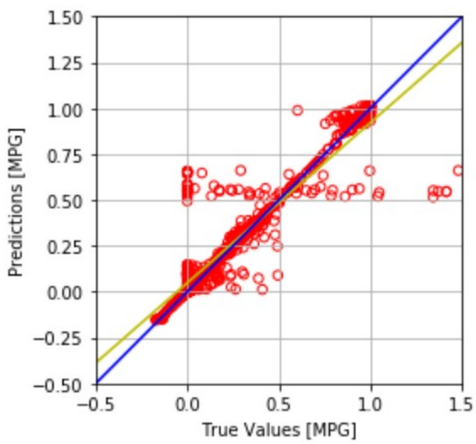

Fig. 8. Test

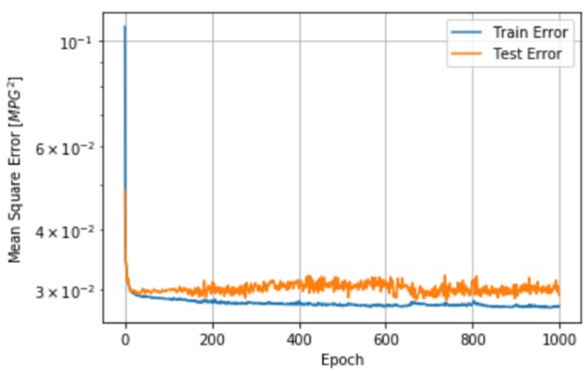

Fig. 9. Mean square error during epochs

In Figure 8 the blue line represents the points for which the predicted values are equal to the real values $(y = 1.0x + 0.0)$ and the yellow line represents the best regression of the points obtained by the code $(y = 0.88x+0.04)$. Pearson's coefficient, which is a parameter that measures linear correlation between X and Y, was also calculated, and the value found was equal to 0.919. Hence, with the straight lines very close and the high correlation, it is possible to say that the model was well trained, despite the existence of some poorly predicted values. A result of the prediction can be seen in Figure 10 which shows a desired airfoil and the one found by the neural network, concluding that the neural network has been trained and validated, despite its limitation in predicting only airfoils of the 4^{th} NACA family.

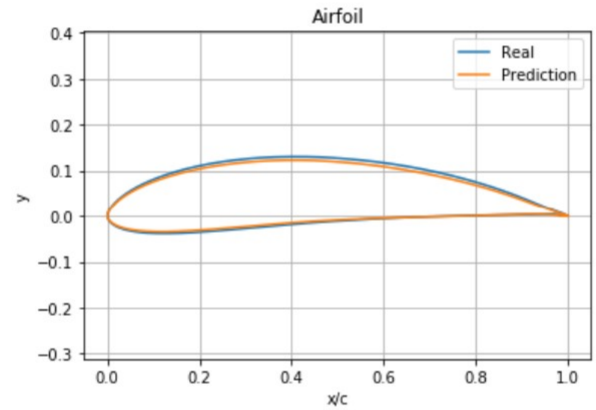

Fig. 10. Prediction exemple

C. Aeroelastic solution

Based on the system described in Section $IV - F$ it was possible to build a structural module, an aerodynamic module and the connection interface between them.

The structural sector is responsible for transforming the loads by applying them at the reference point, in this case the elastic axis, together with the delivery of the displacements from each point of the mesh to the interface. It is also the place where the equation of motion is solved using Euler's numerical integration method. To explain this method, suppose that you want to approximate the solution to a problem of initial value:

$$y'(t) = f(t, y(t)); \quad y(t_0) = y_0 \tag{14}$$

Choosing a value for the time step Δt and assigning each step a point within the range, we have $t_n = t_0 + n\Delta t$. In this, the next step t_{n+1} from the previous t_n is defined as $t_{n+1} = t_n + \Delta t$, so Euler's method consists of determining y_{n+1} via:

$$y_{n+1} = y_n + \Delta t f(t_n, y_n) \tag{15}$$

In which y_n is an approximation of the ODE solution at the point $t_n : y_n \approx y(t_n)$, therefore consisting of an explicit method of integration.

By using this method with the state vector $y = [h \;\; \dot{h} \;\; \theta \;\; \dot{\theta}]^T$ it is possible to obtain an approximation for Equation 12 at each time step Δt. Also using the acceleration values calculated for each iteration described by:

$$\begin{cases} \ddot{h}_t = \frac{I_\theta(F - C_h\dot{h}_t - K_h h_t) - m x_\theta(F - C_\theta\dot{\theta}_t - K_\theta\theta_t)}{m I_\theta - m^2 x_\theta^2} \\ \ddot{\theta}_t = \frac{-m x_\theta(F - C_h\dot{h}_t - K_h h_t) + m(F - C_\theta\dot{\theta}_t - K_\theta\theta_t)}{m I_\theta - m^2 x_\theta^2} \end{cases} \tag{16}$$

It is the responsibility of the aerodynamic module to calculate the loads introduced by the fluid. The code developed by ALTRAN using the 2D panel method was used in this work, basically consisting of a solution technique applicable to the potential flow theory, in which the mathematical equation

TABLE III
CHARACTERISTICS

Mass	$7.0kg$
I_0	$7.0kg.m^2$
X_{cg}	0.4
X_{EA}	0.35
V	$6m/s$
ρ	$1.225kg/m^3$
K_h	$30.0N/m$
K_θ	$50.0N/rad$
C_h	$2.0N/m/s^2$
C_θ	$6.0N/m/s^2$
dt	$0.1s$
Number of steps	$100s$
Airfoil	NACA-3412

governing the method is the Laplace equation, given by the following expression:

$$\bigtriangledown^2\phi = 0 \tag{17}$$

The Equation 17 is valid for a stationary, incompressible and irrotational flow, conditions that are verified in a perfect fluid flow. Although there are several ways to solve this equation, the method used in this case is through singularities, which are algebraic functions that satisfy the Laplace equation. The most common singularities are the sources, the dipoles and the vortices. Since the equation is linear, it is possible to use an overlap of singularities to obtain the solution for a given flow. The complete formulation of the method is described in [1].

Therefore, in possession of the aerodynamic and structural module and the interface, it is possible to carry out simulations in a submerged profile. Performing a simulation for a system with the following characteristics:

The results are shown in Figure 11

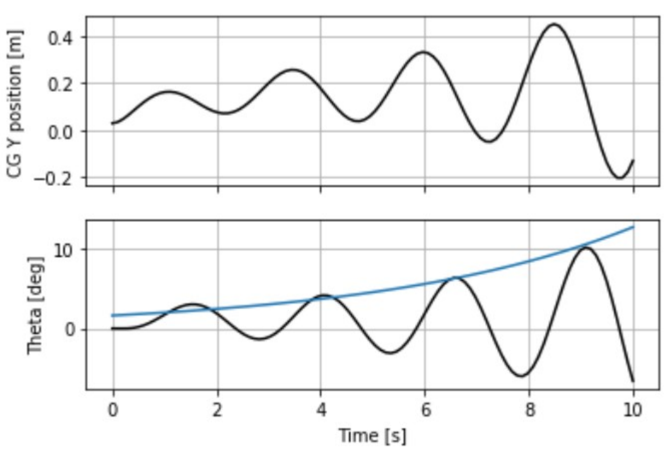

Fig. 11. Aeroelasticity simulation

It is possible to notice the increase in the amplitude of the oscillations over time, therefore, it is identified an unstable flow called Aeroelastic Flutter. It is defined as "an unstable, self-excited structural oscillation at a definite frequency where energy is extracted from the airstream by the motion of the structure" [12]. It represents a type of flow in which the frequencies of the two modes of the system are equal, exciting each other and increasing energy to the flow, which can result in catastrophic situations. The exponential growth envelope of this system can be approximated by an exponential interpolation curve through the signal amplitude peaks, thus used in this work to predict the appearance of instability. The results of this modelling were satisfactory in the system description.

VI. RESULTS OF FINAL COUPLING

Using the described modelling, after all validations, it was possible to couple the codes of the neural network and the fluid-structure, expecting the code to modify the geometry of the airfoil as soon as an instability arises over time. To verify the appearance of instability, the characteristic of its exponential envelope was analysed, whether it was growing or decreasing during the flight simulation. Then, in order to verify the results, the characteristics of a new airfoil were pre-determined using an airfoil of the 4^{th} NACA family, where the neural network is able to predict any airfoils of that family. Additionally, the aim was to alter the profile's center of gravity, which may have been modified in some way (such as jettisoning or fuel consumption) for a region that would cause instability to arise. To make this sudden change in the center of gravity, a change in the positioning of X_{cg} in relation to X_{EA} was made artificially in the simulation at a certain time.

A simulation was performed with the same characteristics as the Table 3, however it started with $X_{cg} = 0.3$ and $X_{EA} = 0.4$, since it is a stable system, as the center of gravity would be in front of the elastic axis according to Pines [16] theory. At eight seconds of real time, the value of X_{cg} was abruptly changed to 0.5, bringing the system to a region of instability and at ten seconds the code was able to detect the increase in energy in the system and the increase in the oscillation amplitudes of the model, thus changing the geometry of the airfoil and advancing the position of the center of gravity, returning the system to a stable position. In Figure 12 the temporal response of the simulation is observed, showing that until eight seconds the amplitudes are reducing in time, in sequence, from eight to ten seconds, the amplitudes start to increase, but after ten seconds (change of geometry and advance of the Cg) the system becomes damped again. To return the system to a stable position, the neural network generated a new airfoil, different from the previous one, which can be seen in Figure 13.

It can be seen that the code was able to predict instability and change the airfoil by correcting and dampening the system.

A problem that was found was the high sensitivity of the neural network when choosing the curves of the desired airfoil. With each simulation in which the neural network was trained, it was possible to obtain different results, which perhaps, were not satisfactory, even with the same training database. This fact occurs due to the random choice of the neural network between

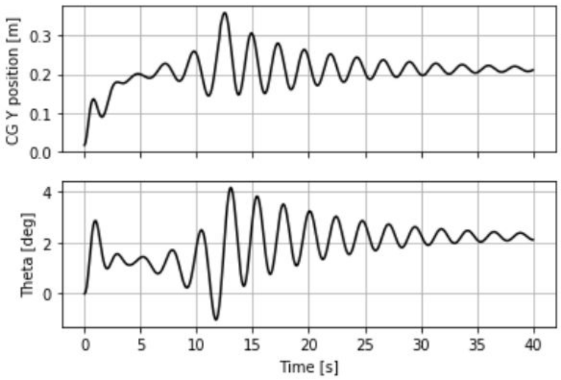

Fig. 12. Time simulation

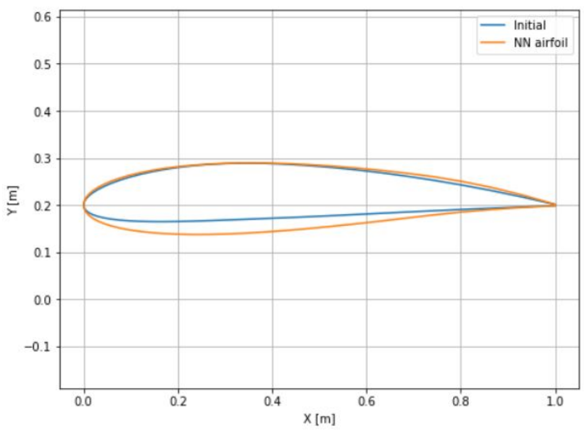

Fig. 13. Airfoil comparison

the training and test data, therefore requiring an increase in the robustness of this network, such as changing its architecture.

VII. Conclusion

This work showed the implementation and validation of methods used in the modelling of an aeroelastic system and a neural network capable of changing the geometry of an airfoil and stabilizing an unstable system. The following tools were used: Bezier-Parsec 3434 parametrization, genetic algorithm, construction of neural network, solution of 2D aeroelastic problem with dynamic alteration of the Cg and final coupling. It is observed through the validation of the codes and the results obtained that the proposed model was well implemented. Figure 12 also shows the correct prevention of the instability that was initiated, thus demonstrating its validity.

However, the problems foreseen in Section II occurred and the greatest difficulty encountered was the construction of the database, a process that required significant time and computational power. The deficiencies of this work were the simplicity of the aeroelastic model, the inability of the neural network to predict different families of airfoils, the sudden change in profile geometry, due to its impossibility in reality, and also the delay in building the database necessary for the training of this network architecture. The main problem was the instability of the neural network's code because it can generate different results depending on the database used for training.

As possible future work, it is necessary to try to correct these problems: the neural network can be improved, making it more generic and powerful; the aeroelastic model and the method of solving the equation of motion can be more accurate and closer to reality. Finally, it is also possible to add new capabilities to the model, such as the introduction of a flap mechanism as a way of controlling instabilities.

References

[1] John D. Anderson. *Fundamentals of aerodynamics*. McGraw-Hill, 5th edition, February 2011.
[2] Alireza Jahangirian Ava Shahrokhi. Airfoil shape parameterization for optimum navier–stokes design with genetic algorithm. *Aerospace Science and Technology 11 443- 450*, 2007.
[3] Jason Bowman, Major Brian Sanders, and Dr. Terrence Weisshaar. Evaluating the impact of morphing technologies on aircraft performance. *43rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, No. AIAA-2002- 1631*, April 2002.
[4] Jason Brownlee. https://machinelearningmastery.com/k-fold-cross-validation/.
[5] François Chollet et al. Keras. https://github.com/fchollet/keras, 2015.
[6] R.W. Derksen and Tim Rogalsky. Bezier-parsec: An optimized aerofoil parameterization for design. *Advances in Engineering Software*, 41:923–930, 2010.
[7] Jimmy Ba Diederik P. Kingma. Adam: A method for stochastic optimization. *3rd International Conference for Learning Representations*, 2015.
[8] Earl H. Dowell. *A Modern Course in Aeroelasticity*. Springer, fifth revised and enlarged edition edition, 1999.
[9] Mark Drela. https://web.mit.edu/drela/Public/web/xfoil/. [Online; accessed on 21 June 2020].
[10] Mohamad H. Hassoun. Fundamentals of artificial neural networks. *MIT press Cambridge, Massachusetts, London*, 1995.
[11] Athar Kharal and Ayman Saleem. Neural networks based airfoil generation for a given cp using bezier–parsec parameterization. *Aerospace Science and Technology*, 23:330–344, 2012.
[12] Rajat Mittal. https://engineering.jhu.edu/fsag/research/aeroelastic-flutter/#:~:text=Aeroelastic%20Flutter,wide%20range%20of%20engineering%20fields.
[13] V. T. Nagaraj and D. J. Johns. Flutter analysis of a wing with concentrated inertias by a direct matrix method. *Symposium on structural dynamics paper No. E.5*, 1975.
[14] Nakahashi K. Oyama A, Obayashi S. Fractional factorial design of genetic coding for aerodynamic optimization. *AIAA Paper 99 - 3298*, 1999.
[15] D.P. Raymer. Aircraft design: A conceptual approach. *4th edn., AIAA Education Series, AIAA*, 2006.
[16] Pines S. An elementary explanation of the flutter mechanism. *Proceedings nature specialists meeting on dynamics and aeroelasticity, Institute of the Aeronautical Sciences, Ft.Worth, Texas, pp 52–58*, 1958.
[17] Nilesh P. Salunke, Juned Ahamad R. A., and S.A. Channiwala. Airfoil parameterization techniques: A review. *American Journal of Mechanical Engineering*, 2:99–102, no. 4 (2014).
[18] Michael S. Selig. http://www.ae.illinois.edu/m-selig/ads/coord_database.html. [Online; accessed on 20 May 2020].
[19] R. Storn. http://www.icsi.berkeley.edu/~storn/code.html. [Online; accessed on 01 May 2020].
[20] Rogalsky T. Acceleration of differential evolution for aerodynamic design. ph.d. thesis. *University of Manitoba*, 2004.
[21] N. Tsushima, K. Soneda, T. Yokozeki, T. Imamura, H. Arizono, and W. Su. Structural and aerodynamic models for aeroelastic analysis of corrugated morphing wings. *AIAA SciTech Forum*, Jan 2020.
[22] Andreas Zell. *Simulation Neuronaler Netze [Simulation of Neural Networks] (in German)*. Addison-Wesley. p. 73. ISBN 3-89319-554-8, 1 edition, 1994.
[23] Andreas Zell. Simulation neuronaler netze [simulation of neural networks] (in german) (1st ed.). *Addison-Wesley.Chapter 5.2.*, 2003.