POLITECNICO DI MILANO

Scuola di Ingegneria Industriale e dell'Informazione

Corso di Laurea in Space Engineering

Optimization of variable stiffness composite plates with Artificial Neural Network

Relatore: Prof. Riccardo Vescovini

Correlatore: Dott. Alessio Gorgeri

Tesi di Laurea di:

Federico PORTA Matr. 896865

Anno Accademico 2019-2020

Acknowledgements

I wish to thank my supervisors, Professor R. Vescovini and PhD student A. Gorgeri, for their support, helpfulness, and encouragement. Their effort and time They dedicated to this work were extremely precious to me. The helps that they gave me during the work reported in this thesis was fundamental to complete the work. Furthermore, I am grateful to Professor R. Vescovini, for the possibility He gave me to work on my graduation thesis under His supervision on one of the most interesting subjects.

I wish to thank also all of my friends and my family. This work is dedicated to them, that gave me fundamental support. They have always encouraged me, believing in my possibilities and in my abilities.

Abstract

Every day new strategies are studied to reduce the computational cost and the time consumption for the preliminary design phase of composite material structures. Surrogate models and optimization algorithms that are no gradient based are particularly interesting due to their efficiency with a relatively small consumption of resources. The goal of this work is to develop a computational tool based on Artificial Neural Networks for analysing structural problems. The study is focused on free vibration and buckling problems of variables stiffness panels (VSP) maximizing the value of the first natural frequency and the first buckling load, respectively. Furthermore, in the analysis is introduced a parametrization of the orientation angles of the fibres by using Lamination Parameters. However, this conversion of the design variables introduces non-linear constraints that requires some attention during sampling procedures. One crucial aspect associated with the use of Lamination Parameters is the need for recovering a set of orientation angles if the laminate that is optimized would be manufactured. The maximization of the considered values is obtained implementing a particle swarm optimization (PSO) algorithm inside the process. Furthermore, the PSO is used also to obtain the angle orientation from the lamination parameters. All the computational parts are done using MATLAB® with the use of codes written for this study and ones already implemented in the software. The work demonstrates that the process studied halves significatively the time consumption respect classical methods, introducing in the process an error of maximum 3%.

Keywords: variable stiffness panels, lamination parameters, artificial neural network, particle swarm optimization, transfer-learning.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

When a structural part is designed, preliminary analysis are conducted to evaluate the effects of material, thickness, or other characteristic on the desired performances. Different methods are commonly used. The most common strategy refers to the use of the finite element analysis (FEM). However, the time consumption can be relatively high for preliminary design studies. New strategies are thus of interest to reduce the time spent and the computational costs, while guaranteeing similar level of prediction accuracy respect the standard strategies. One of the possibilities is that of using a surrogate model, i.e. a data-driven model that mimics the physics of the problem.

Goal of this investigation is the analysis of variable-stiffness plates (VSP) obtained by means of curvilinear fibre paths. Different schemes were proposed in the literature [1][2] for the orientation of the fibres for this kind of composite structures, increasing the complexity of the problem as the distribution is more elaborate. In fact, where the path of the fibres is not straight, a series of parameters are added to fully describe the laminate.

To simplify the problem reducing the number of design variables, a parametrization is implemented by Sethoodeh et al. [3][4] and by IJsselmuiden [5]. Therefore, the orientation angles of the fibres are converted in a set of lamination parameters. However, this conversion has some drawbacks. A post-processing is needed to evaluate the orientation angles distribution in each ply related to the lamination parameters sets. Another drawback that is introduced by the parametrization, is the need of a different failure criterion that does not depend on the orientation angles.

Due to the large number of degrees of freedom of VSP structures, non gradient-based optimization methods are preferable for a more efficient analysis [6]-[9]. The particle swarm optimization algorithm (PSO) or the genetic algorithm (GA) are examples of non gradient-based algorithms.

The work focuses on the evaluation and maximization of the first natural frequency and the first buckling load.

1.1 Variable stiffness laminates

Composite laminates can be subdivided into two main categories depending on the scheme used to place the fibres in the plies. The first category refers to the classical laminate fibres orientation, where each ply has a constant orientation angles along the ply. The second category is composed by the variable stiffness composites (VSP). This kind of composite structure can be manufacturable due to the innovation on the fibre placement that allows a curvilinear displacement of the fibres.

Variable stiffness structures have different patterns of orientation angles of the fibres, that are placed with a linear or with a greater order of variation along one or both axes [1][2]. As a reference, different patterns are reported in the following figure:

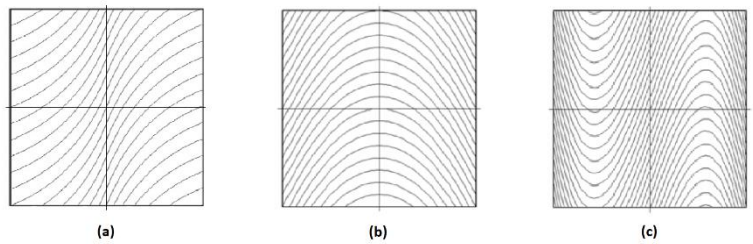

Figure 1: Curvilinear fibre path along one axis in a ply that varies linearly (a), quadratically (b), and cubically (c) [1]

This kind of composite structure are characterised by a larger amount of degrees of freedom respect to the classical laminate structure, i.e. a quadratic distribution along both axes leads to 9 variables for each ply. The degrees of freedom obtained in this way allow a more efficient design obtaining structures with improved properties.

Gürdal et al. [10] considered the orientation angles of the fibres in some defined points of the structure, obtaining the angles between them by interpolation. However, due to the augmented complexity, in literature parametrization of the orientation angles are commonly used in order to reduce the number of design variables that have to be tuned [3]-[5].

The advantages of such strategies were expressed by Pasini et al. [2] comparing the properties of classical and variable stiffness plate. In fact, the VSP structures allow a better distribution of the stiffness of the plate. For example, the application of span-wise constant membrane load might results in a non-constant membrane deformation [11], as reported in the figure below:

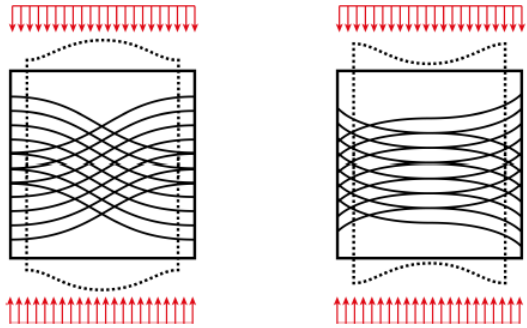

Figure 2: Example of VSP where the dash lines represent the deformation related to the load applied [11]

The deformations shown in Figure 2 are due to the misalignment of the orientation of the fibres respect to the direction of the load. The structural benefits of VSP are obtained by tailoring the material properties in directions that are more favourable to carry loads within the laminates. Liu et al. [12] and Gürdal et al. [13] demonstrated a better resistance to instability of the VSP structures. Furthermore, Khashaba et al. [14] shown the possibility to build VSP with even the presence of one or more holes.

1.2 Manufacture of composite materials

The technologies behind the manufacture of composite materials change and evolve during years. The classical composite structure can be built by manual placement of each ply or even with the use of mechanical processes, i.e. robotic harms. Robotic harms were used in filament winding and the implementation of complex algorithms inside those ones grant the possibility to build more elaborate patterns of the fibres with an excellent precision. Furthermore, more complex shapes of structures can be built with these robotic harms, reducing the efforts, and saving time. In figure below is reported an example of automated fibre placement machine:

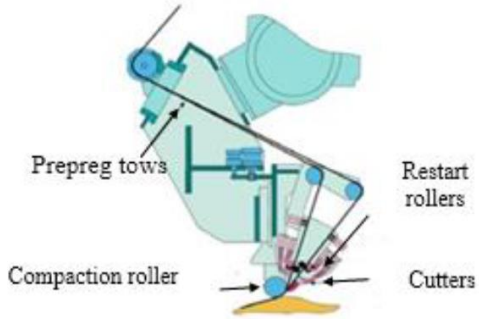

Figure 3: Automated fibre placement machine [15]

The placement head of the fibre placement machine can soften the prepreg tows and places it on the surface of the mold according to the predetermined path, and then presses it with the compaction roller. The tow contains many fibres and several of these generate the tow bands. Each tow can be cut and resent independently, and the angle of the tow can be changed. Furthermore, the placed fibres have to be bent with a radius bigger than the maximum radius of curvature allowed for the fibre, in order to avoid their cracks during the placement procedure or avoid wrong placement due to the mechanical properties of the material. Zheng et al. [15] studied methods to avoid gap and defect locations in the placement of

the tows due to the cutting of the fibres. The properties of the gap and defected areas are not the one obtained from the project design, generating regions where the stiffness drops, or cracks can happen more easily. A graphical representation of the two different areas is reported in figure:

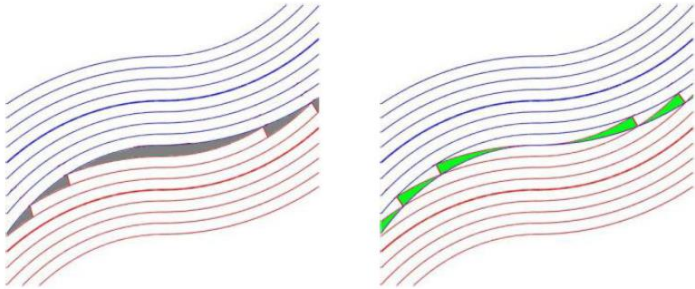

Figure 4: Example of gap region in grey (left), and completed gap regions obtaining defected areas in green (right) [15]

The manufacture disposition of the fibres can also modify the buckling load capability [15] because the overlapped parts act as ribs, increasing the range of loads that can be applied.

1.3 Lamination parameters

Lamination parameters are employed in substitution of the orientation angles. Classical structural optimization problems can be solved with the use of lamination parameters [16][17] and also for problems with variables stiffness composite structures [3]-[5].

The lamination parameters can be used to obtain the lamination constitutive matrices of the classical lamination theory, using material invariants and reduced stiffness components. Furthermore, lamination parameters can define the shear terms of the laminate constitutive matrices and the ones referred to the thermal load [5][16]. However, lamination parameters must verify some non-linear

constrains, introduced by their definition, in order to have a physical meaning [18]-[20].

The use of a VSP increases the number of parameters that have to be taken into account during the whole optimization process, reflecting on the complexity of the surrogate model and on the optimization algorithm. The increase of the number of degrees of freedom is balanced by the reduction apported by the conversion because, instead of a large number of orientation angles, only a small set of numbers is sufficient to fully describe the laminate. In fact, lamination parameters condense all the angles in a set of 12 values, independently from the number of plies [3]-[5][18]. However, as the complexity of the orientation of the fibres increase in the VSP, more parameters are needed to properly describe the laminate configuration. For example, if the variation of fibre orientation is at most linear along a single direction, 2 sets of lamination parameters are needed.

One of the main advantages of introducing lamination parameters in optimization processes, it is that the optimization surface become convex, allowing a more efficient convergence to the optimal value, as reported by Setoodeh et al. [3].

The use of lamination parameters requires a conversion procedure after the optimization process to define the orientation angles of the laminate and make the manufacture feasible. There exist different strategies to convert back the lamination parameters to the orientation angles. Among the others, optimization algorithms, as studied by Friswell et al. [21], or conversion with the use of stream function, as reported by Setoodeh et al. [4], can be used. The process to obtain the orientation angles from the lamination parameters is affected by errors depending on the complexity of the algorithm used, that varies on the number of variables considered. The design variables of the post-processing conversion can span over different characteristics of the material and the structure, i.e. orientation of the fibres, curvature radius (for VSP), thickness of the ply, differences between the angles of one ply with the following one.

The lamination parameters can also be used in the context of failure analysis. However, classical methods reported in literature [22] implemented criteria that need the orientation angles to obtain the strain distribution in each ply in order to check if failures occur. Therefore, a new strategy has to be introduced to check if failure occurs in the laminate using directly the parametrization.

1.4 Failure analysis

Failure analysis is often conducted, at least in the early design steps, referring to a first ply failure criterion. Among the many criteria available in the literature, the maximum stress criterion, the maximum strain criterion, the Tsai-Hill criterion, the Hoffmann criterion, and the Tsai-Wu criterion [22] are the most commonly implemented. All the aforelisted criteria are used to determine if a structure can withstand a certain load, depending on the orientation angles of the fibres. However, if the orientation angles are substituted by lamination parameters, these criteria need to be reformulated accordingly.

A criterion that can be applied irrespective of the stacking sequence is needed. For instance, IJsselmuiden et al. [5][23] presented a solution that solve this kind of problem. The strategy implements a Tsai-Wu failure criterion analysis that use the lamination parameters to check every possible associated orientation of the angles - not only the one presented in the structure considered as in the classical methods - and verifies if a failure occurs. The modified Tsai-Wu method presented refers to a first-ply-failure criterion. This implies that the laminate is considered damaged when the first failure appears in a ply of the structure.

1.5 Surrogate models

Surrogate models are used to reduce time consumption and computational costs while guaranteeing a reasonable level of accuracy. Different types of surrogate models exist, i.e. Polynomial Regression (PR), Radial Basis Function (RBF), Artificial Neural Networks (ANN), Kriging (KRG), and Support Vector Regression (SVR) [24][25]. Each surrogate model is composed by polynomials or functions that approximate the interconnection between the input data and the output. Pasini et al. [2] consider some surrogate models (not the ANN), i.e. the “Polynomial Regression (PR)”, the “Radial Basis Function (RBF)”, the “Kriging KRG)”, and the “Support Vector Regression (SVR)”. From this analysis, it is determined that the best performance for VSP are obtained using KRG algorithm.

Another promising surrogate method present in the literature, that is not analysed by Pasini et al. [2], is the Artificial Neural Network. This method is widely employed in literature to solve structural problems due to its versatility and efficiency. Bisagni et al. [6] optimize the post-buckling properties of composite stiffened panels using

ANN, determining not only the number of plies with orientation $\pm 45°$, but also the number and the side dimension of the stiffeners. Cardozo et al. [7] minimized the cost and weight of an in-plane loaded composite laminated plates, maximized the stiffness shell, and maximized the first natural frequency of a laminated plate. Ruijter et al. [8] used ANN to predict the strain and buckling multipliers for composite panels with the presence of holes and stiffeners. Shakeri et al. [9] investigated the stacking sequence optimization of laminated cylindrical panels with the first natural frequency as the objective function. Cardoso and Shakeri demonstrated that the procedure that involves ANN is comparable in performance with the ones that implements nearly exact numerical computations. They demonstrated that ANN has a small difference in accuracy, but also shown a reduction in the computational time consumption, using the surrogate model instead of the numerical one.

ANN are also employed in fields that do not involve structural problems thanks to their capability of manipulate a large amount of data and their ability to generalize result. Some examples of ANN implementation are about time series prediction with trend and seasonal behaviour [26][27], research of mathematical function to be employed inside ANN [28], detection of leaks in gas pipelines with acoustic method [29], optimization of chemical process (industrial cracking furnace) [30], optimization of the performance of an aircraft wing modifying the geometric parameters [31], and micro-mechanical models approximation to detect the mechanical properties of carbon nanotubes fibres [32].

Artificial neural networks are composed by a series of neurons organized in different layers, that are interconnected one with the other with various topologies [33][34]. In literature multilayer perceptron (MLP) network are largely employed [2][6]-[9][26][27][29][31][32] with different architectures like “bridge networks”, “cascade networks”, or “Radial Basis Function networks (RBFn)” [29]. For tasks that could be affected by noise, the RBFn is the best architecture to obtain small errors [29][35][36], but generally the other topologies of MLP have smaller errors [35][36]. The ANN has to be designed in order to have the smallest number of neurons possible. This has to be done to avoid losses in generality, maintaining a good approximation level [33], and this task can be done also using automata algorithm [37]. Each neuron has inside an activation function to elaborate the input data and obtaining an output. For what concerns this functions, Karlik et al. [38], and Zhang et al. [39] investigated the performances of different activation functions, increasing even more the degree of freedom that have to be tuned to obtain the most efficient ANN for the every specific problem. Furthermore, Zoph et al. [28]

studied a method to use artificial neural networks to optimise the activation function inside the neurons, building new function with the use of an ANN.

Artificial neural networks need to be trained to perform their tasks by using sets of input-output data. The sets are obtained with observation measurements or by numerical programmes, such as finite element analysis (FEM) or Ritz's analysis. Therefore, if the sets are obtained by observation measurements, the number of sets that are available is fixed, otherwise this number can be treated as a design variable. In all the studies in the literature reported, FEM is used to obtain data to train the ANN. For what concerns the training, there exist different algorithms and even more strategies that modify these methods in order to make them faster or more accurate, i.e. changing some internal parameters [40]-[42], increasing the order of the algorithm [34][43] and implementing a reduced matrix to speed up the process [44][45]. Even optimization algorithms can be used to train an ANN [46][47], or to find the optimal number of neurons that have to be considered inside the ANN itself [30][48][49].

The ANN has also to be tested in order to verify its accuracy, using sets of input-output data that are not used for the training process. The error on the test sets rates the level of approximation, allowing a comparison between different training strategies and compositions of the network [34].

Generally, surrogate models are implemented in optimization procedures as objective function to solve computational heavy problems that require a huge amount of time to obtain the results. These methods grant a good level of approximation as demonstrated by Pasini et al. [2], Queipo et al. [24], and Hamadi et al. [25].

1.6 Optimization methods

Structural optimizations are often conducted during the early design steps and, for this scope, different optimization algorithms are available.

There exist different strategies of algorithms such as the "Simplex method" (Dantziy 1947), "branch and bound method" [17], "Newton method", "feasible direction method", "Prim's algorithm" and "Kruskal's algorithm" (for tree problems). A wide class of techniques refers to evolutionary algorithms. This kind of optimization algorithms tries to mimic the nature like insect swarm (particle swarm optimization), ant colonies (ant colony optimization) or even chromosome

crossover and breeding (genetic algorithm optimization). Both “particle swarm optimization algorithm (PSO)” [11][26][46][50] and “genetic algorithm (GA)” [6]-[9] were implemented in structural optimization. This gradient free optimization techniques requires a large number of evaluations. Therefore, the strategies that implements this kind of algorithms are useful if the analysis takes a small amount of time. Furthermore, Venter et al. [46] and Ghashochi Bargh et al. [50] demonstrated that for structural problem the PSO algorithm could have better performance with respect to the others.

Some of the internal parameters of the optimization algorithms must be tuned to allow a global convergence. In fact, depending on the starting point the optimum search can be trapped in a local global, that it could be mistaken for the optimal value of the desired characteristic. There exist algorithms, such as the PSO, where the starting point does not affect the convergence of the optimization process [46]. However, even for this optimization algorithm, the possibility to remain trapped in local optimum exists.

Another aspect that has to be taken into account is the boundary condition of the design variables. Most of the algorithms cannot consider complex constraints, but only simple upper and lower bound on the value of the variables [46]. Therefore, penalty terms must be introduced inside the objective function [11][21], allowing the introduction of linear and non-linear constraints, and even Boolean ones.

1.7 Aim of the study

The aim of the study is to develop a procedure to solve structural optimization problems with reduced CPU efforts. Two example problems are proposed to fulfil this task, where the aim is to find the optimal value for the first natural frequency and the first buckling load for a VSP structure.

Lamination parameters are implemented to reduce the number of design variables. To obtain the starting data to train the ANN, a Ritz’s analysis is executed.

The artificial neural network is selected as surrogate model due to reduce the CPU efforts with a small loss in the accuracy. Two separated networks are trained, one for the buckling problem and the other one for the free vibration problem. In addition, the transfer learning strategy is implemented to further reduce the time consumptions. However, the efficiency of this method has to be demonstrated.

Classical ANNs are to be preferred over RBF networks due to the better performances for problems where noiseless data are available, as reported by Pasini et al. [2].

The optimization is solved by implementing in the process the PSO. Furthermore, this algorithm is also used in this work to solve the post-processing conversion to obtain the orientation angles to make the manufacture feasible.

In addition, a failure criterion to check if the laminate described with the use of lamination parameters suffers cracks under the action of a specific load applied is introduced to complete the design process.

The configuration that implements lamination parameters, ANN, and PSO is not used all together in literature. In addition, smaller innovations are present in some parts of the work, i.e. the transfer-learning analysis to connect the buckling and the free vibrational problem, and the weights determination for the PSO objective function in the post-processing phase. In this latter case, the strain energy [51] is used to determine the weights of the different parts inside the objective function. This allows to select the weights in order to keep some properties that could be lost if they are selected in other ways.

1.8 Outline

The thesis is structured as follow:

- In Chapter 2 the Ritz's analysis method is briefly explained. The number of the function used, and the number of integration points are reported with motivations. Then a validation of the numerical method is done for both the problem considered in order to check the accuracy of the algorithms implemented.

- The lamination parameters conversion is reported in Chapter 3. The difference between the evaluation process of the stiffness tensors with the use of the lamination parameters and the orientation angles of the fibres is reported. Then the sampling method implemented to obtain the training and testing set for the ANN is shown. Furthermore, the failure criterion is expressed, reporting all the formulas. Finally, the data sets

obtained for solve the example problems are reported, with also a final validation section of the implemented failure analysis.

- In Chapter 4 the artificial neural network parts are explained. The different ANN architectures and the possible activation function that can be selected for the neurons are shown. The most used training algorithm are reported with all the formulas. Furthermore, how the training sets are obtained, and the explanation of the testing part are presented. Then the transfer-learning strategy is explained, and finally all the design process results are reported. Finally, validations are done in order to verify the efficiency of the ANN obtained.

- In Chapter 5 the PSO algorithm is explained. A list of different methods to implement constraints checks and the method to add penalty terms depending on the needs are shown. How the ANN find place inside the optimization algorithm is briefly explained, also reporting some parts of the MATLAB® code. Then the methods chosen to how determine some internal variables values are explained. Finally, the results of the optimization process are reported and compared with the ones outputted from a Ritz's analysis. This is done to evaluate the efficiency of the PSO.

- The post-processing procedure that converts the lamination parameters in the orientation angles of the fibres is reported in Chapter 6. The method based on the strain energy to determine the objective function weights is shown. Similar for what done for the PSO, some internal parameters have been tuned. Furthermore, a validation is done to demonstrate that the code executes the conversion properly. Then the final conversion results, with the relative errors, and the values of the internal variables are reported.

- The numerical results for two different example problems are reported in Chapter 7 (free vibration problem and buckling problem).

- In Chapter 8 the final considerations and some future developments of the procedure presented in this study are expressed.

Chapter 2

Ritz's method

In this chapter the Ritz's method analysis that focus on VSP is explained. This kind of numerical analysis will be used to obtain all the data needed for the input-output data sets used in future part of this work. Furthermore, it allows also to evaluate average load, stress, and strains that can be used to determine some weights parameters for the post-processing procedure.

2.1 Ritz's method and example problems resolution process

The Ritz's method transforms the partial differential equations of the structural problem in systems of algebraic equations. The method relies on approximating the unknown displacement field as a truncated summation of known shape functions and unknown amplitudes. For instance, the out of plane displacement of a plate is expressed as:

$$w(x,y,t) = \sum_{i=1}^{T} \phi_i(x,y) u_i(t) \tag{2.1}$$

where w is the displacement along the z-axis, T represent the total number of degrees of freedom considered ($i = 1{:}T$ with $T = R \text{ x } S$), $\phi_i(x,y)$ is the i-th Ritz's function depending on the position on the plate (x,y), while $u_i(t)$ is the generalized coordinate or Ritz's amplitude referred to the i-th degree of freedom. The amplitudes are the unknows that have to be determined in order to obtain the displacements. The equation (3.1) can be generalized to the other 5 degrees of freedom (displacements and rotation), introducing also prescribed displacement along the plate boundaries. The 6 equations can be put in vector form as reported in equation (2.2).

$$\underline{d_0} = \begin{bmatrix} \boldsymbol{\phi_u} & 0 \\ 0 & \boldsymbol{\phi_\varphi} \end{bmatrix} \begin{Bmatrix} \underline{a_u} \\ \underline{a_\varphi} \end{Bmatrix} + \begin{Bmatrix} \overline{\boldsymbol{\phi_u}} \\ 0 \end{Bmatrix} = \boldsymbol{\Phi}\underline{a} + \bar{\boldsymbol{\Phi}} \tag{2.2}$$

where $\underline{d_0}$ is the vector that collects the generalized displacement components $\underline{u_o}$ and $\underline{\varphi}$ of the kinematic model along the three axes, $\underline{a}$ is the vector of the Ritz unknown amplitudes, $\boldsymbol{\phi_u}$ and $\boldsymbol{\phi_\varphi}$ are the matrices containing the column vectors of dimension R_i x S_i collecting the trial functions for the generalized displacement and generalized curvature, respectively [52].

In this case, the Ritz's functions can be defined as a combination of two functions along different axis as reported in the equation below:

$$\phi_i(x, y) = X_m(x)Y_n(y) \tag{2.3}$$

The Ritz's functions can be polynomial or trigonometric, considering that the accuracy of the solution depends on the kind of function considered, and on the order of expansion. The convergence of a Ritz set is guaranteed if and only if the hypotheses of completeness and admissibility are satisfied. They have to be mathematically complete (no subscripts miss in the set of functions), the essential condition must be satisfied with a continuity order of C^{n-1} where n is the maximum order of the derivative present in the variational principle. If the Ritz's functions set is an admissible one the convergence is granted.

If the Ritz's approximation is implemented inside the Principle of Virtual Works (PVW), and integrating the known form functions, is obtained an equation as reported in equation (2.4). It has to be highlighted that the values of the Ritz's function and of the value of the mechanical properties of the plate has to be evaluated in the integration points, where their number must be larger than the set of the functions.

$$\boldsymbol{M}(x, y)\underline{\ddot{u}}(t) + \boldsymbol{K}(x, y)\underline{u}(t) = \underline{F}(x, y, t) \tag{2.4}$$

where $\boldsymbol{M}(x,y)$ and $\boldsymbol{K}(x,y)$ are the mass matrix and the stiffness matrix of the plate, respectively, while $\underline{F}(x,y,t)$ is the total applied load vector. It has to be underlined that the mass and stiffness matrices are composed by the integration on the plate surface of the partial differential Ritz's functions multiplied by the mass per unit surface and the stiffness laminate constitutive matrices, respectively.

To determine the first natural frequency a free vibrational problem has to be solved. Hypothesizing that the solution is in the form of:

$$\underline{u}(t) = \underline{U_0} e^{i\omega t} \tag{2.5}$$

The problem is reduced in the form of:

$$[-\omega^2 \boldsymbol{M}(x,y) + \boldsymbol{K}(x,y)]\underline{U_0} = \underline{0} \tag{2.6}$$

To avoid trivial solution, the terms in the square parenthesis has to define a null determinant. This can be achieved solving an eigenvalue problem:

$$\det\left(-\omega^2 \boldsymbol{M}(x,y) + \boldsymbol{K}(x,y)\right) = 0 \tag{2.7}$$

The eigenvalues obtained in this way are the squared natural frequencies of the plate (ω^2), while the eigenvectors are the amplitudes that are used to reconstruct the vibration modes. Furthermore, the smaller natural frequency (ω) is defined "first natural frequency".

For what concern the buckling problem, the non-linear terms of the deformation have to be taken into account in order to consider the coupling between the in-plane loads and the bending moments. Considering the deformation with order of magnitude greater than 1, and applying the Trefftz's criterion, two sets of equation are obtained: the pre-buckling and the buckling equations. These equations must be solved to obtain the first buckling load. The pre-buckling problem is analysed

solving a static problem, while the buckling one is composed by an eigenvalue problem. The smaller eigenvalue ($\min(\lambda)$) is the multiplicator value to obtain the first buckling load, as reported in the following equation:

$$(\mathbf{K} - \lambda \mathbf{K_\sigma})\underline{u}(t) = \underline{0} \quad \rightarrow \quad N_{xx_{buck}} = min|\lambda|\widehat{N_{xx}} \tag{2.8}$$

The matrices $\mathbf{K}$ and $\mathbf{K_\sigma}$ are the stiffness matrix independent from the pre-buckling and the geometric stiffness matrix that contain the pre-buckling stresses, respectively. The $\widehat{N_{xx}}$ is the applied load that has to be multiplied by $\min(\lambda)$ in order to obtain the exact value of the buckling load. The stiffness matrix ($\mathbf{K}$) is evaluated with the use of the laminate constitutive matrices. The geometric stiffness matrix ($\mathbf{K_\sigma}$) is obtained from the linearization of the non-linear terms of the strain-displacement relation accordingly with the Von Kármán assumptions. Using a Ritz's approximation, the geometric stiffness matrix is obtained as:

$$\mathbf{K_\sigma} = \int_{-1}^{1}\int_{-1}^{1}(\boldsymbol{\mathcal{B}_2}\boldsymbol{\Phi})^T \begin{bmatrix} N_{xx}(\xi,\eta) & N_{xy}(\xi,\eta) \\ N_{xy}(\xi,\eta) & N_{yy}(\xi,\eta) \end{bmatrix} \boldsymbol{\mathcal{B}_2}\boldsymbol{\Phi}\mathbf{J}\, d\xi\, d\eta \tag{2.9}$$

where $\boldsymbol{\mathcal{B}_2}$ is a differential matrix, the pre-buckling force resultants N_{ik} are determined with an initial pre-buckling analysis based on an energy approach, and the terms ξ and η are the nondimensional coordinates $(\xi,\eta) \in$ [-1 , 1] [52]. In particular, $\xi = 2/a$ and $\eta = 2/b$ where a and b are the plate dimensions. The pre-buckling analysis is executed with a linear static analysis on the laminate. The differential matrix introduced in equation (2.9) is expressed as:

$$\boldsymbol{\mathcal{B}_2} = \begin{bmatrix} 0 & 0 & (\cdot)_{,x} & 0 & 0 & 0 \\ 0 & 0 & (\cdot)_{,y} & 0 & 0 & 0 \end{bmatrix} \tag{2.10}$$

For variables stiffness laminate the assembled matrices are fully populated because of the numerical integration process. In the particular case of composite panels with straight fiber orientation, the Ritz's integrals can be carried out analytically, increasing the degree of sparsity of the matrices, reducing the time required for the solution.

2.2 Validation and internal parameters values

The Ritz's method is coded in MATLAB® and must be validated to grant that the numerical results obtained are correct. The code implemented in the software is the same used by Vescovini et al. [52]. Two separate validations are done, one for the constant stiffness laminate and the other one for the VS plate.

The first one is subdivided in free vibration and buckling problem. For the free vibration problem, the Ritz's implemented code is validated comparing the results obtained with the ones reported by Crawley [53]. The problem to be solved is a cantilever composite plate, with dimension 76 mm x 76 mm x 1.04 mm where the nominal ply thickness is 0.13 mm. The plate is composed by 8 plies made of AS/3501-6 graphite/epoxy (material properties data are taken form [53]). The results obtained are reported in the table below:

Laminate	Observed Freq. (Hz)	Calculated Freq. (Hz)	% Diff.
$[0_2/\pm30]_s$	234.2	256.5	8.7
$[0/\pm45/90]_s$	196.4	219.5	10.5
$[\pm45/\mp45]_s$	131.2	135.1	2.9

Table 1: First natural frequency comparison for validation of the Ritz's method

To solve the free vibrational problem, different number of Ritz's function and integration points are tested. The smaller number of these parameters that grants the convergence is 12 Ritz's function both along x and y, with 13 integration points both along x and y and 27 points along z (thickness). The obtained percentage errors

reported in Table 3 are in the order of 10% or even less, obtaining a good approximation of the first natural frequencies.

For what concern the buckling problem, the validation is done analytically, resolving manually some simple cases and comparing the $\min(\lambda)$ value obtained. Also in this case, three different laminate are considered with dimensions equal to 1000 mm x 1000 mm x 1.04 mm. The plates are simple supported and composed by 8 plies made of AS/3501-6 graphite/epoxy. The results obtained are reported in the following table:

Laminate	Analytical λ	Calculated λ	% Diff.
$[0_2/90_2]_s$	0.0272	0.0272	0
$[0/90/\pm45]_s$	0.0412	0.0412	0
$[\pm45/\mp45]_s$	0.2004	0.2004	0

Table 2: Multiplicator for the load applied to the plate to obtain the first buckling load for validation of the Ritz's method

In the buckling load problem, the analytical and the calculated solutions match perfectly. The values of the parameters are the same one determined for the free vibrational problem: 12 Ritz's function along x and y, with 13 integration points along x and y and 27 point along z (thickness).

The Ritz's method code used is validated also for VSP configurations. The buckling problem expressed in literature by Vescovini et al. [52] is considered, and the same validation procedure is done. The nondimensional buckling parameters $K_{cr} = N_{xx}^{cr} \frac{a^2}{E_{11} t^3}$ for two different simply supported symmetric square plates of side dimension 254 mm under prescribed axial shortening are reported. The elastic properties of the material considered are reported in Table 3.

	Material Properties
E_{11}	181.000 MPa
E_{22}	10.270 MPa
G_{12}	7170 MPa
G_{13}	4000 MPa
G_{23}	4000 MPa
ν_{12}	0.28
ρ	1.35E-9 t/mm^3

Table 3: Material thermo-elastic properties for VSP Ritz's method validation

The two laminate are composed by 8 layers each with a thickness per play equal to 0.127 mm. The Ritz's analysis is executed with 15 functions along x and y, with 20 integration points along x and y and 35 along z (thickness). Only the first 4 modes are considered and the fixed displacement along x is equal to 0.0016 mm in compression. The results obtained with the Ritz's code implemented are compared with the ones reported by Wu et al. [54]. The results for the first two mode are reported in the table below for each configuration considered:

Laminate	Mode n.	Ritz [54]	Ritz	% Diff.
A	1	3.4991	3.5209	0.6
	2	3.5026	3.5250	0.6
B	1	3.7112	3.6865	0.7
	2	3.7227	3.7075	0.4

Table 4: Nondimensional buckling parameter K_{cr} for simply supported VSP under prescribed axial shortening

The laminate A and B are composed by non-linear distribution of the angles of the form $[\pm\vartheta_1/\pm\vartheta_2]_s$, where ϑ_i are vectors or matrices containing the angles in 3 or 9 points on the surface of the plate, respectively. The mesh of these points is generated by a grid that subdivides equally the two half dimension of the plate. The two vectors for the laminate configurations are reported in Table 5.

	Laminate A	Laminate B
ϑ_1	$[68 \quad 55 \quad 19]$	$\begin{bmatrix} 71 & 49.5 & 71.5 \\ 67 & 50 & 51 \\ 17 & 12 & 45 \end{bmatrix}$
ϑ_2	$[-76 \quad -55 \quad 9]$	$\begin{bmatrix} -72.5 & -59 & -59.5 \\ -65 & -54 & -50.5 \\ 14 & 11.5 & 6 \end{bmatrix}$

Table 5: Laminate configurations A and B of table 4

The pre-buckling membrane resultants N_{xx}, N_{yy}, and N_{xy} for the laminate B are reported in the figure below:

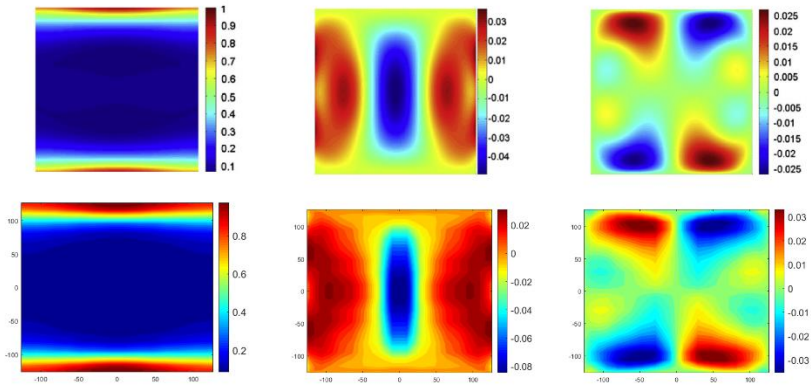

Figure 5: Pre-buckling membrane resultants for VSP loaded in compression (N_{xx} left, N_{yy} center, and N_{xy} right)(first row [54], second row implemented Ritz)

In Figure 5 the first row represents the pre-buckling membrane resultants obtained in literature [54], while the second row is the resultants obtained with the Ritz's method that is implemented in the software.

Observing the results obtained for both the validation processes, the Ritz’s method analysis implemented can be used to solve numerically structural problems. In fact, the percentage error founded during the validation process are only in the order of 10% for the free vibration problem, while is under the 1% for the buckling one. This level of accuracy can be considered for a preliminary design phase.

Chapter 3

Lamination Parameters

Lamination parameters are introduced in this chapter. This parametrization of the orientation angles changes the set of design variables of the problem, describing the composite structure with a smaller number of variables. The lamination parameters are non-dimensional quantities that transform the laminate design parameters into a set of 12 values. Therefore, this new parametrization is independent from the number of the plies, reducing the complexity of the problem for structure with very elaborate fibres orientation variation. However, the implementation of this conversion implies that the laminate constitutive matrices have to be expressed in function of the lamination parameters.

The use of lamination parameters in place of ply angles is often preferable in optimization problems. In this latter case the objective function, that has to be maximised, is often non-convex.

Another problem that is introduced with the implementation of this conversion is the failure analysis. A new strategy must be introduced to check if failure happens even without converting the lamination parameters in angles. Therefore, the Tsai-Wu failure criterion is modified.

3.1 Lamination parameters definition

The lamination parameters are defined from the ply angles by integrating trigonometric functions along the normalized thickness. Considering a reference frame fixed in the center of the plate surface and on the middle plane, the mathematical transformation from angles to lamination parameters is expressed in the following equations (3.1) [4][5][18].

$$\begin{aligned}
\xi^{A}_{[1,2,3,4]} &= \frac{1}{2}\int_{-1}^{1}[\,\cos 2\vartheta(\bar{z})\,,\ \sin 2\vartheta(\bar{z})\,,\ \cos 4\vartheta(\bar{z})\,,\ \sin 4\vartheta(\bar{z})\,]\ d\,\bar{z} \\
\xi^{B}_{[1,2,3,4]} &= \int_{-1}^{1}[\,\cos 2\vartheta(\bar{z})\,,\ \sin 2\vartheta(\bar{z})\,,\ \cos 4\vartheta(\bar{z})\,,\ \sin 4\vartheta(\bar{z})\,]\,\bar{z}\ d\,\bar{z} \\
\xi^{D}_{[1,2,3,4]} &= \frac{3}{2}\int_{-1}^{1}[\,\cos 2\vartheta(\bar{z})\,,\ \sin 2\vartheta(\bar{z})\,,\ \cos 4\vartheta(\bar{z})\,,\ \sin 4\vartheta(\bar{z})\,]\,\bar{z}^2\ d\,\bar{z}
\end{aligned} \tag{3.1}$$

where $\vartheta(\bar{z})$ is the distribution function of the ply orientation angles through normalized thickness coordinate $\bar{z} = (2/h)z$, where h is the thickness of the laminate and z is the coordinate along the normal. The **A**-**B**-**D** superscript are associated with the in-plane, coupling, and out-of-plane laminate constitutive matrices, respectively. This definition follows the nomenclature introduced by the classical laminate plate theory (Tsai & Hahn - 1980) reported in equation below [5][22]:

$$\begin{Bmatrix} \underline{N} \\ \underline{M} \end{Bmatrix} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{B} & \boldsymbol{D} \end{bmatrix} \begin{Bmatrix} \underline{\varepsilon}^{0} \\ \underline{k} \end{Bmatrix} \tag{3.2}$$

where $\underline{N}$ is a vector of resultant membrane loads, $\underline{M}$ is a vector of resultant out-of-plane moments, $\underline{\varepsilon}^{\mathbf{0}}$ is the vector of mid-plane strains, and $\underline{k}$ is the vector of plate curvatures.

For straight fiber laminates, only one set of 12 lamination parameters can fully describe the laminate constitutive matrices (four for each matrix). However, when the orientation angles vary along the in-plane directions, as in the case of Variables Stiffness Panels (VSP), the 12 lamination parameters are also function of the in-plane position. The lamination parameters contain all the information needed to determine the elastic properties of a laminate, therefore they can be used in place of ply orientation as design variables.

3.2 Laminate constitutive matrices determination

The laminate constitutive matrices reported in equation (3.2) have to be evaluated in order to solve structural problems. With the classical definition of the problem,

where the orientation angles are given, the components of the three matrices are obtained as [22]:

$$\begin{aligned} A_{ij} &= \sum_{k=1}^{N} Q_{ij_{(k)}}(z_k - z_{k-1}) \\ B_{ij} &= \frac{1}{2}\sum_{k=1}^{N} Q_{ij_{(k)}}(z_k^2 - z_{k-1}^2) \qquad (i = 1,2,6) \\ D_{ij} &= \frac{1}{3}\sum_{k=1}^{N} Q_{ij_{(k)}}(z_k^3 - z_{k-1}^3) \end{aligned} \tag{3.3}$$

where N is the number of layers in the laminate, $Q_{ij_{(k)}}$ are reduced stiffness for unidirectional lamina of layer k, and $t_k = z_k - z_{k-1}$ is the thickness of the k-th layer. The Q_{ij} terms can be computed as reported [16][18][22]:

$$\begin{aligned} Q_{11} &= E_{11}^2/(E_{11} - E_{22}\nu_{12}^2) \\ Q_{22} &= E_{11}E_{22}/(E_{11} - E_{22}\nu_{12}^2) \\ Q_{12} &= \nu_{12}Q_{22} \\ Q_{66} &= G_{12} \end{aligned} \tag{3.4}$$

$$\begin{aligned} Q_{44} &= G_{23} \\ Q_{55} &= G_{31} \end{aligned} \tag{3.5}$$

where E_{11}, E_{22}, G_{12}, G_{23} and G_{31} are the longitudinal, transverse and shear moduli, respectively, while ν_{12} is the Poisson's ratio for a unidirectional laminate. It can be observed that the reduced stiffness terms are dependent only on material properties. However, the direction 1, 2, and 3 used as subscripts are referred to the principal orientation of the material, without defining the orientation of the fibres with respect to the frame of reference. To fully describe a layer, the angles are needed in order to make a rotation of the values in order to align them with the frame of reference of the laminate.

The components of the tensors **A**, **B** and **D** could be defined also with linear functions of lamination parameters and material invariants [5][16][18], as expressed in equation (3.6-9).

$$\begin{pmatrix} A_{11} \\ A_{22} \\ A_{12} \\ A_{66} \\ A_{16} \\ A_{26} \end{pmatrix} = h \begin{bmatrix} 1 & \xi_1^A & \xi_3^A & 0 & 0 \\ 1 & -\xi_1^A & \xi_3^A & 0 & 0 \\ 0 & 0 & -\xi_3^A & 1 & 0 \\ 0 & 0 & -\xi_3^A & 0 & 1 \\ 0 & \xi_2^A/2 & \xi_4^A & 0 & 0 \\ 0 & \xi_2^A/2 & -\xi_4^A & 0 & 0 \end{bmatrix} \begin{pmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \\ U_5 \end{pmatrix} \tag{3.6}$$

$$\begin{pmatrix} B_{11} \\ B_{22} \\ B_{12} \\ B_{66} \\ B_{16} \\ B_{26} \end{pmatrix} = \frac{h^2}{4} \begin{bmatrix} 0 & \xi_1^B & \xi_3^B & 0 & 0 \\ 0 & -\xi_1^B & \xi_3^B & 0 & 0 \\ 0 & 0 & -\xi_3^B & 0 & 0 \\ 0 & 0 & -\xi_3^B & 0 & 0 \\ 0 & \xi_2^B/2 & \xi_4^B & 0 & 0 \\ 0 & \xi_2^B/2 & -\xi_4^B & 0 & 0 \end{bmatrix} \begin{pmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \\ U_5 \end{pmatrix} \tag{3.7}$$

$$\begin{pmatrix} D_{11} \\ D_{22} \\ D_{12} \\ D_{66} \\ D_{16} \\ D_{26} \end{pmatrix} = \frac{h^3}{12} \begin{bmatrix} 1 & \xi_1^D & \xi_3^D & 0 & 0 \\ 1 & -\xi_1^D & \xi_3^D & 0 & 0 \\ 0 & 0 & -\xi_3^D & 1 & 0 \\ 0 & 0 & -\xi_3^D & 0 & 1 \\ 0 & \xi_2^D/2 & \xi_4^D & 0 & 0 \\ 0 & \xi_2^D/2 & -\xi_4^D & 0 & 0 \end{bmatrix} \begin{pmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \\ U_5 \end{pmatrix} \tag{3.8}$$

$$\begin{pmatrix} A_{44} \\ A_{45} \\ A_{55} \end{pmatrix} = h \begin{bmatrix} 1 & \xi_1^A \\ 1 & -\xi_1^A \\ 0 & -\xi_2^A \end{bmatrix} \begin{pmatrix} U_1' \\ U_2' \end{pmatrix} \tag{3.9}$$

Additionally, the material invariants are defined in equation (3.10) [16][18][22].

$$\begin{aligned} U_1 &= [3Q_{11} + 3Q_{22} + 2Q_{12} + 4Q_{66}]/8 \\ U_2 &= [Q_{11} - Q_{22}]/2 \\ U_3 &= [Q_{11} + Q_{22} - 2Q_{12} - 4Q_{66}]/8 \\ U_4 &= [Q_{11} + Q_{22} + 6Q_{12} - 4Q_{66}]/8 \\ U_5 &= [Q_{11} + Q_{22} - 2Q_{12} + 4Q_{66}]/8 \end{aligned} \tag{3.10}$$

$$\begin{aligned} U_1' &= 1/2Q_{44} + 1/2Q_{55} \\ U_2' &= 1/2Q_{44} - 1/2Q_{55} \end{aligned} \tag{3.11}$$

According to equations (3.4 - 11), the material properties and the total thickness of the laminate are the only quantities involved in the computation of the laminate composite matrices.

Through the lamination parameters it is also possible to evaluate the thermal membrane load and moment stress vectors resultants [5]:

$$\begin{aligned} \underline{N}^{Th} &= h(\underline{\Lambda}_0 + \xi_1^A \underline{\Lambda}_1 + \xi_2^A \underline{\Lambda}_2)\Delta T \\ \underline{M}^{Th} &= \frac{h^2}{4}\left(\xi_1^B \underline{\Lambda}_1 + \xi_2^B \underline{\Lambda}_2\right)\Delta T \end{aligned} \tag{3.12}$$

where ΔT is the applied temperature difference that induces the thermal strains and $\underline{\Lambda}_i$ are vectors defined as:

$$\begin{aligned} \underline{\Lambda}_0 &= (\alpha_1 Q_{11} + (\alpha_1 + \alpha_2)Q_{12} + \alpha_2 Q_{22}) \cdot \{1 \quad 1 \quad 0\}^T \\ \underline{\Lambda}_1 &= (\alpha_1 Q_{11} + (\alpha_1 - \alpha_2)Q_{12} - \alpha_2 Q_{22}) \cdot \{1 \quad -1 \quad 0\}^T \\ \underline{\Lambda}_2 &= (\alpha_1 Q_{11} + (\alpha_1 - \alpha_2)Q_{12} - \alpha_2 Q_{22}) \cdot \{0 \quad 0 \quad 1\}^T \end{aligned} \tag{3.13}$$

where α_1 and α_2 are the coefficients of thermal expansion along primary material directions, and Q_{ij} are the reduced lamina stiffness components, that are reported in equation (3.4).

3.3 Constraints

Although each lamination parameter can take values between -1 and +1, not every combination is admissible as the trigonometric function used in equation (3.1) are not independent each other. Diaconu et al. [19] derived the whole set of constraints of the twelve lamination parameters, implementing the variational approach developed by Grenestedt and Gudmundson in 1993 [20]. The feasible domain for the in-plane lamination parameters is defined as [16][18][19]:

$$\begin{gathered} 2{\xi_1^A}^2(1-\xi_3^A)+2{\xi_2^A}^2(1+\xi_2^A)+{\xi_3^A}^2+{\xi_4^A}^2-4\xi_1^A\xi_2^A\xi_4^A \;\leq\; 1 \\ {\xi_1^A}^2+\;{\xi_2^A}^2\;\leq\;1 \\ -1\;\leq\;\xi_i^A\;\leq 1 \qquad (i=1,\dots,4) \end{gathered} \tag{3.14}$$

An identical set of expression can be obtained for the out-of-plane lamination parameters, maintaining the same formula of equation (3.14) and the same subscripts, but changing the superscripts (from ξ_i^A to ξ_i^D). Furthermore, additional inequality constraints can be derived between certain sets of in-plane and out-of-plane lamination parameters [18][19]:

$$\tfrac{1}{4}(\xi_i^A+1)^3-1\;\leq\;\xi_i^D\;\leq\;\tfrac{1}{4}(\xi_i^A+1)^3+1 \qquad (i=1,\dots,4) \tag{3.15}$$

The feasible region of lamination parameters of the coupling laminate constitutive matrix can be obtained with the inverse of the relations that link the in-plane, coupling, and out-of-plane terms, as expressed in equation (3.16). The indices of the parameters in the formulas are the same for each term as reported below [18][19]:

$$\begin{aligned} 4(\xi_i^A+1)(\xi_i^D+1)\;&\geq\;(\xi_i^A+1)^4+3(\xi_i^B)^2 \\ 4(\xi_i^A-1)(\xi_i^D-1)\;&\geq\;(\xi_i^A-1)^4+3(\xi_i^B)^2 \end{aligned} \qquad (i=1,\dots,4) \tag{3.16}$$

These constraints have to be verified for each set of lamination parameters describing the laminate.

3.4 Latin Hypercube Sample (LHS)

The Latin Hypercube Sample (LHS) strategy is implemented in order to introduce a rigorous method to sample properly the lamination parameters sets.

The LHS is a statistical method with a stratified sampling approach that generates a near-random sets of values from a multidimensional space [24]. Stratified sampling ensures that all portions of a given partition are sampled. The algorithm consists in dividing the n-dimensions, each representing a design variable, in m equally probable intervals, where m is the number of sample points to be generated. The m points are chosen one by one in order to cover as much as possible the feasible design space. Therefore, only one point can belong to a single interval for all the dimensions, as can be observed for the highlighted point in Figure 6.

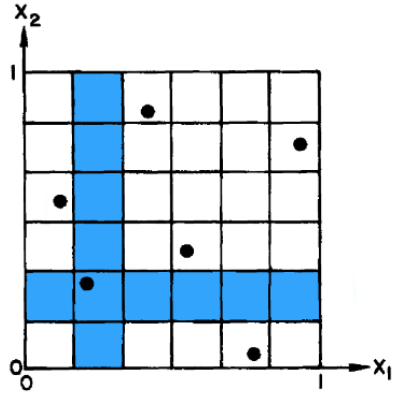

Figure 6: LHS design with n=2, m=6 for X uniformly distributed on the unit square

With this procedure, the possible combination of values that the sample points can assume reduces step by step. However, the advantage of this iteration step strategy

consists in generating a complete set, where in each interval of the design variables one point is certainly sampled.

The Latin Hypercube Sampling method can provide sampling plans with very different performance in terms of uniformity, that differ by measured, for example, the minimum distance among design points. In the figure below the difference between two sampling plans can be observed:

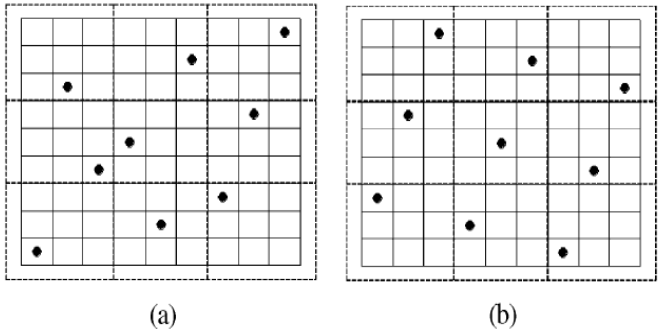

Figure 7: LHS design with differences in terms of uniformity

It can be noted that the design of the LHS in Figure 7 (b) is better than the one reported in (a). In fact, the (b) distribution maps uniformly the design space, without generating clusters and too large holes.

MATLAB® is used to sample the lamination parameters in the design space. The Latin Hypercube Sampling algorithm implemented in MATLAB® generates a continuous design space between 0 and 1. However, lamination parameters are defined between -1 and 1, as shown by the definition of section 3.1 and the constraints expressed in section 3.3. The sets sampled by the LHS must be scaled in order to fit the right design space.

The MATLAB® implemented Latin Hypercube is not able to check if non-linear constraints existing between the lamination parameters terms are verified.

Therefore, to overcome this obstacle, a filtering process is done to reduce the sampled point only to the ones that verify the constraints listed in section 3.3.

The scheme of the filtering process is reported in the figure below:

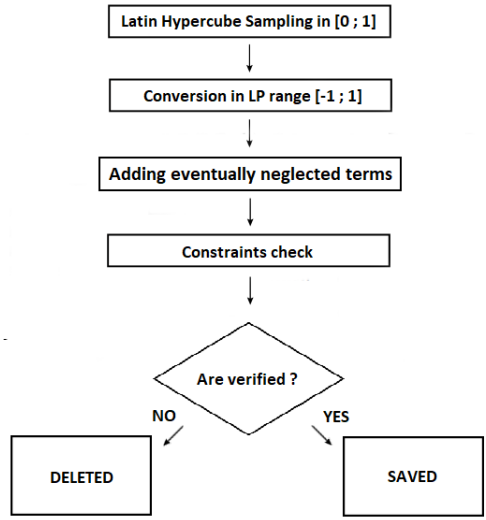

Figure 8: Filtering process logic

If assumptions are introduced in the considered problem, some terms could be neglected because will be null. Therefore, before the constraints are checked, the lamination parameter set has to be completed, reintroducing inside the set all the neglected terms, allowing it to enter the filtering process.

The filtering process starts checking all the augmented LP sets. Only the sets that respect the constraints are retained, while the others are discarded.

The full procedure of the sampling can be summarized in the following steps:

1) Define a 380 times larger number of points in respect to the desired one to be sampled in a twelve-dimensional space bounded between -1 and 1.
2) Adding terms eventually neglected for problem simplifications or assumptions.
3) Check if in every point needed to fully describe the laminate, the lamination parameters sets verify the constraints in equations (3.14 - 16).
4) Collect and store only the sets of lamination parameters that respect point 3.

In Figure 9 an example of sampling distribution is reported. The boundary reported in equation (3.14) and equation (3.15) are expressed with a solid line. The lamination parameters terms are indicated as V_i^k instead of ξ_i^k where *i* is linked to the terms (1, 2, 3 and 4) and *k* is related to the lamination constitutive matrix (**A**, **B**, or **D**).

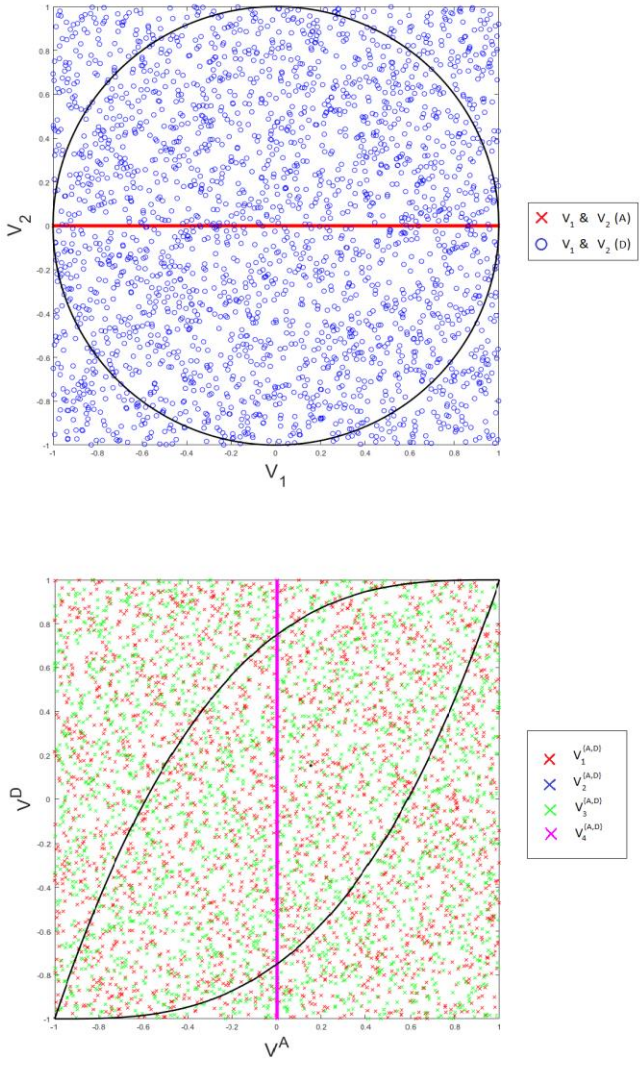

Figure 9: Sample points distribution ($V_i^k = \xi_i^k$) for second equation (3.14)(up), and for equation (3.15)(bottom)

Considering that a huge number of sampling points does not verify the constraints, because the lamination parameters sets are generated on all the space between -1 and 1, the number m that has to be selected needs to be larger than the desired one. This can be observed in Figure 9, where the lamination parameters sets and the non-linear limit curves are reported. All the points inside the limit curves verify the constraints and are saved. Furthermore, not all the points that are inside one limit curve verify the constraints, because those points could be outside other limit curves referred to other constraints. In fact, between the 2500 points sampled by the LHS and reported in Figure 9, only 6 verify all the constraints. The number of starting points sampled with the LHS before the filtering process needs to be 380 times larger than the number of desired points that are wanted as an output.

With the filtering process, the spatial distribution of sets is not guaranteed to fill the design space uniformly, which may lead to a suboptimal mapping. This can be deducted also from the example reported in Figure 9, where the 2500 points map the entire space uniformly, but the resultants 6 points that are saved cannot describe the design space properly.

3.5 Tsai-Wu failure criterion

In this section a modified Tsai-Wu failure criterion is presented, where lamination parameters are used in place of orientation angles for checking the failure. As the classical failure analysis criterion, the modified Tsai-Wu criterion approaches the problem with the first-ply-failure strategy. The first ply failure is widely used in the analysis of composite structures [5][22][23].

The standard approach relies on the orientation angles for each ply or, for VSP, in all the points needed to describe the laminate. For both these kinds of structure, the analysis is executed in various points due to the possible non-homogeneous distribution of the stress field.

When the problem is formulated in terms of lamination parameters, one first possibility consists in converting lamination parameters to orientation angles. However, this procedure is not free of computational errors. This is due to the fact that, unlike the conversion from angles to LP, the inverse transformation cannot be expressed in closed-form. It can be noted that, as the number of plies increases, the conversion error gets lower. This is due to the fact that more physical variables are

available to match the lamination parameters. However, when a relatively small number of plies is considered, the reconstruction error can be non-negligible.

A more adequate approach for handling problems formulated in terms of LP consists in rephrasing the criterion in a more suitable way. Specifically, with the modified criterion, the failure can be studied knowing the material properties of the laminate, its strength properties, and the applied and internal strains. An advantage of the modified criterion over the classical one relies in considering all the possible orientation of the angles, generalizing the problem [5][23]. This does not need the explicit transformation of the lamination parameters, avoiding in this way the conversion errors, working directly on the parameters.

The development of the modified criterion is presented, starting from the failure envelope of the classical Tsai-Wu failure criterion, which is given by [5][22]:

$$F_{11}\sigma_1^2 + F_{22}\sigma_2^2 + F_{66}\tau_{12}^2 + F_1\sigma_1 + F_2\sigma_2 + 2F_{12}\sigma_1\sigma_2 = 1 \tag{3.17}$$

where F_i and F_{ij} are the second- and fourth-order strength tensors, with $i, j = 1, 2, 6$ as reported in equation below:

$$\begin{aligned} F_{11} &= \frac{1}{X_t X_c} & F_{22} &= \frac{1}{Y_t Y_c} & F_1 &= \frac{1}{X_t} - \frac{1}{X_c} \\ F_2 &= \frac{1}{Y_t} - \frac{1}{Y_c} & F_{12} &= \frac{-1}{2\sqrt{X_t X_c Y_t Y_c}} & F_{66} &= \frac{1}{S^2} \end{aligned} \tag{3.18}$$

where X_t, X_c, Y_t, Y_c, and S are the failure strength in compression, tension, and shear in the principal material direction.

The failure criterion can be reformulated in terms of the components of the material strain tensor, expressed as:

$$G_{11}\epsilon_1^2 + G_{22}\epsilon_2^2 + G_{66}\epsilon_{12}^2 + G_1\epsilon_1 + G_2\epsilon_2 + 2G_{12}\epsilon_1\epsilon_2 = 1 \tag{3.19}$$

where G_{ij} are the strain coefficients, while the material strains ($\epsilon_1, \epsilon_2, \epsilon_{12}$) can subsequently be related to the laminate strains ($\epsilon_x, \epsilon_y, \epsilon_{xy}$) using a transformation matrix. This matrix contains sine and cosine functions associated with the angles of the ply. The strain coefficients are defined as:

$$\begin{gathered} G_{11} = Q_{11}^2 F_{11} + Q_{12}^2 F_{22} + 2F_{12}Q_{11}Q_{12} \qquad G_1 = Q_{11}F_1 + Q_{12}F_2 \\ G_{22} = Q_{12}^2 F_{11} + Q_{22}^2 F_{22} + 2F_{12}Q_{12}Q_{22} \qquad G_2 = Q_{12}F_1 + Q_{22}F_2 \\ G_{!2} = Q_{11}Q_{12}F_{11} + Q_{!2}Q_{22}F_{22} + F_{12}Q_{12}^2 + F_{12}Q_{11}Q_{22} \\ G_{66} = 4Q_{66}^2 F_{66} \end{gathered} \tag{3.20}$$

where the Q_{ij} are reduced stiffness indicated in equation (3.4 - 5).

The transformation matrix ($\boldsymbol{R}$) used to convert the material strains into laminate strain is reported below [5][22][23]:

$$\boldsymbol{R} = \begin{bmatrix} \frac{1}{2}(1+\mathrm{c}) & \frac{1}{2}(1-c) & s \\ \frac{1}{2}(1-c) & \frac{1}{2}(1+c) & -s \\ -\frac{1}{2}s & \frac{1}{2}s & c \end{bmatrix} \tag{3.21}$$

where $s = \sin(2\vartheta)$ and $c = \cos(2\vartheta)$.

From equation (3.19) the geometric "envelope" is constructed. It is defined as the surface tangent to the family of failure surfaces, which depends on the laminate strains and the ply angles. However, parametrizing with the use of the ply angles, the geometric envelope can be defined regardless of the ply orientation. This parametrization is obtained eliminating all the sine and cosine terms introduced by the rotation matrix $\boldsymbol{R}$, this is achieved by using Dixon's resultant for the elimination of polynomial equations [23]. The resultant equations that represent the boundaries surfaces of the envelope of the failure criterion for all ply orientations are expressed in equations (3.22-23) [5][23].

$$4u_6^2 I_2^2 - 4u_6 u_1 I_2^2 + 4(1 - u_2 I_1 - u_3 I_1^2)(u_! - u_6) + (u_4 + u_5 I_1)^2 = 0 \quad (3.22)$$

$$u_1^2 I_2^4 - I_2^2 (u_4 + u_5 I_1)^2 - 2u_1 I_2^2 (1 - u_2 I_1 - u_3 I_1^2) + (1 - u_2 I_1 - u_3 I_1^2)^2 = 0 \quad (3.23)$$

where I_1 is the volumetric strain invariant and I_2 is the maximum shear strain defined as:

$$I_1 = \epsilon_x + \epsilon_y \qquad I_2 = \sqrt{\left(\frac{\epsilon_x - \epsilon_y}{2}\right)^2 + \epsilon_{xy}^2} \quad (3.24)$$

The terms u_i $(i = 1, \dots, 6)$ are expressed in terms of the strain coefficients of the equation (3.20) as:

$$\begin{aligned} u_1 &= G_{11} + G_{22} - 2G_{12} & u_2 &= (G_1 + G_2)/2 \\ u_3 &= (G_{11} + G_{22} + 2G_{12})/4 & u_4 &= G_1 - G_2 \\ u_5 &= G_{11} - G_{22} & u_6 &= G_{66} \end{aligned} \quad (3.25)$$

The feasible space inside the two surfaces, drawn by the equations (3.22) and (3.23), is material dependent. In fact, the u_i terms are in function of strain coefficients that are in turn a function of the reduced stiffness matrix and material strength coefficients. The envelope of equation (3.22) is defined by a second-order equation with respect the strains, while the second envelope expressed by the equation (3.23) is defined by a curve of the fourth-order. These two envelope equations do not intersect one another but may become tangent. The inner region enclosed within the two envelopes represents the region in which no failure occurs. Therefore, the laminate strains must fall inside the envelope to avoid failure (according to the first-ply failure criterion).

As a reference, two different envelopes referred to two different laminate are reported in Figure 10. The left one is referred to a Carbon-Epoxy (IM6) single ply, where on the right are reported the various curves associated with a Boron-Epoxy (B5.6) single ply plate [23]. The Tsai-Wu method that implements the lamination parameters and the classical one are compared, considering that ε_{xy} is set to zero. Therefore, to show the generalization properties of the new method, a set of different orientation angles is considered. It can be observed that all of these lines are inside the envelope generated by the modified method.

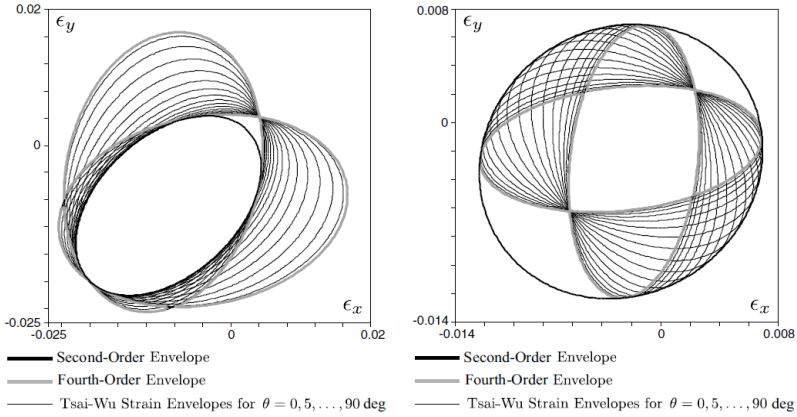

Figure 10: Modified Tsai-Wu criterion strain envelopes [23]

The method illustrated herein can only predict if the laminate undergoes failure in a point of a ply in the stacking sequence for an applied load.

A validation is done referring to the example reported in [22]. In the example a simply supported $[0_2/\pm 45]_s$ Kevlar/Epoxy laminate is loaded by biaxial loads with ratio $N_y/N_x = 0.5$. The range of loads evaluated along the x-axis that is admitted for this kind of laminate is in between $-161 \cdot 10^3$ N/m and $93.6 \cdot 10^3$ N/m. The modified failure criterion is used, and the results describe a new range of value for the load admitted along the x-axis. The load needs to be in between $-60 \cdot 10^3$ N/m and $60 \cdot 10^3$ N/m. As expected by the theory expressed in the literature [5][23], this new criterion generalizes the failure analysis. The new admissible domain of

values is reduced. This is due to the fact that all the orientation angles are considered, introducing in the analysis also laminates with a worse response to this kind of loads configuration.

Chapter 4

Artificial Neural Network (ANN)

In this chapter the procedures required to build an artificial neural network (ANN) are presented. The neural network is implemented as a surrogate model in order to allow some benchmark problems to be solved with improved efficiency. The artificial neural network is coded in MATLAB®.

The ANN is a very powerful instrument that can be used every time an interconnection between the inputs and the desired output exists. The type of interconnection could be different, including classification of data, physical phenomena, economical, personal flavours, and so on. For structural problems, the ANN aims at reproducing the mechanical behaviour by weighting and combining the inputs of the problem to output the desired quantities.

The neural network tries to mimic the human brain. As the human brain processes the information obtained from the sensors (eyes, hears, skin, tongue, etc.), the ANN takes the information given in input, and with a mathematical reworking inside nodes, generates the output. The skeleton of the surrogate model is composed by nodes that work as neurons. The nodes are connected one with the others following certain schemes that can have large variation in terms of number of layers and neurons.

The network needs to be trained to fulfil its task. The training part is the core of the procedure inasmuch a poor training leads to poor results. The training of the network is one of the most complex and time expensive part of the process. Different methods may be used to tune the internal parameters of the ANN. Furthermore, in order to reduce the time spent in the training process of the network, a transfer-learning strategy is briefly introduced. This training method can be used when more than one ANN is used, trying to connect the training procedures for all the network to the first trained one.

4.1 Artificial Neural Network

The Artificial Neural Network is composed by different elements that characterise the network itself. Each one will be explained in this section, that it is structured as follow:

1. ANN elements.
2. Neurons distribution.
3. Network architectures.
4. Activation functions.

4.1.1 ANN elements

As a reference, the elements inside an artificial neural network are shown in the figure below [29][34][44]:

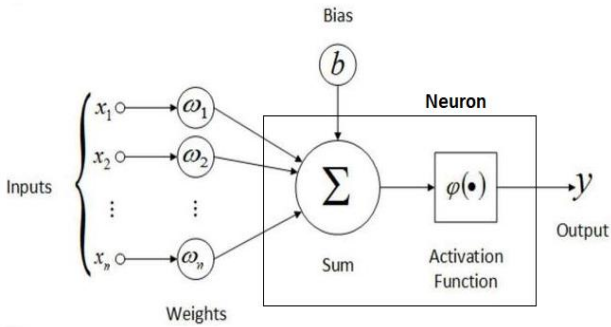

Figure 11: Elements of an artificial neural network

The elements that build an ANN are divided into weights, bias, inputs, output, and activation function. Some of the elements are inside neurons, others connect the neurons, as shown in Figure 11.

The flow of information enters in the left side and exits from the right side. Therefore, the inputs are weighted and summed with the bias; then the summation pass through the activation function generating the output of the neuron. The inputs can be outputs of other neurons, input data introduced from other computational procedure, or a combination of the two. The bias consists in a unitary input that is weighted as a normal input.

The activation function and the nature of the input that enters inside a specific neuron are selected a priori ad remain fixed during the training procedure. There exist also many activation functions and each neuron could have its own function that differs from the other present in the network.

Furthermore, the network can be organized with different architectures, varying the connection between the layers. The number of weights increases as the network become more complex, introducing more interconnection between different networks. The weights are determined with a tuning procedure, named training process. The training process can be based on different algorithms that have their advantages and disadvantages, granting different level of efficiency.

4.1.2 Neurons distribution

The network is characterized by assembling different layers composed by neurons that can be subdivided in three categories based on their characteristics [29][44]:

a) **Input layer**: there is only one input layer in the network, and it is placed at the beginning of the information flow. The input layer contains all the inputs x_i where i ranges from 1 to n. It is important to underline that no upper bound exists for the value of n, although it should be noted that the complexity of the problem increases as n gets higher. Therefore, more and more complex networks are required to accurately predict the desired outputs. Note that the number of inputs is often given a-priori, i.e. it is not a design variable, unlike the number of neurons in the hidden layers.

b) **Hidden layer**: this layer is the internal part of the network and can be considered the core of the ANN. The number of these layers can be different depending on the problem complexity, down to a minimum of a single layer. The hidden layers are composed by groups of neurons put

in parallel where their number could vary layer by layer. The type of interconnection between one hidden layer with the other ones determines the architecture of the network. The interconnection between hidden layers consists of considering as the input for a layer the output values that exit from the previous one.

c) **Output layer**: this is the last layer that closes the network. This layer has as many neurons as the number of outputs, and each neuron returns one single value. The number of outputs is, in principle, not bounded. However, the complexity of the problem increases as the number of the desired outputs increase.

A classical interconnection between the three different type of layers is shown:

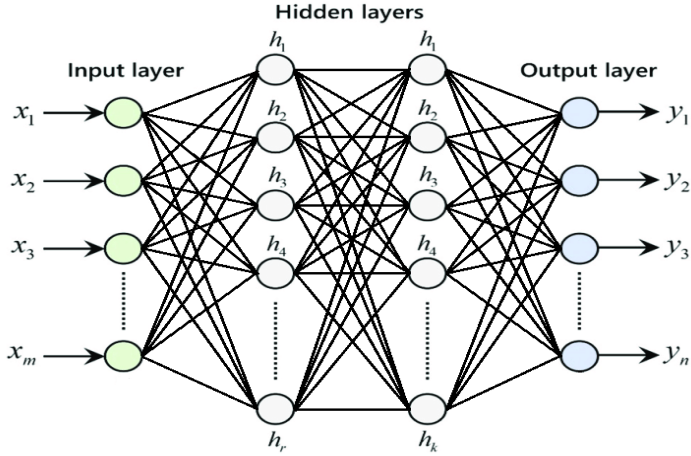

Figure 12: Interconnection between the three kinds of layers in the artificial neural network

where h_i is the i-th neuron in the hidden layer and y_t is the t-th output where t ranges from 1 to n. The circles represent the different neurons and the line are the interconnection between them, where each of them is associated with a weight.

4.1.3 Network architectures

The topology of the interconnection of the hidden layers neurons defines the type of architecture of the artificial neural network. Several architectures exist, the most popular ones being the "*feedforward* network", the "*cascade* network", the "*bridge* network", and the "*radial basis function* network" [33][35][36][43]:

a) **Feedforward network**: this is one of the simplest topologies for an artificial neural network. Each hidden layer is connected with the previous and the following ones only. The feedforward type reduces the complexity of the network without dragging the input information through the process. Many problems in engineering applications can be solved with this kind of network considering as few as 2 hidden layers. A graphical representation of this architecture is shown in Figure 12.

b) **Cascade network**: this topology connects each layer not only to the immediately following one but also to all subsequent layers. A graphical representation of this type of network is shown in the following figure where only one neuron is considered in each hidden layer in order to simplify the scheme:

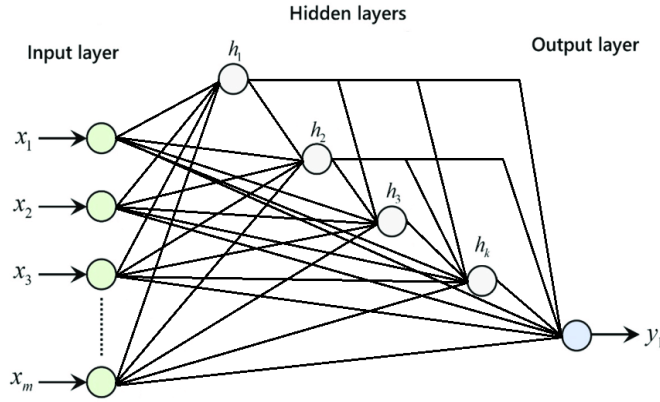

Figure 13: Interconnection scheme for a cascade network

In this way a cascade of information is built because inputs or even the results obtained by neurons are considered in each neuron, without "losing" such information. The cascade configuration implies a large number of weights, thus increasing the complexity of the network. For this reason, the training for this architecture is more time expensive. A problem that could be solved by a simple feedforward network can also be solved by a cascade network, although the vice versa is not guaranteed.
It should be pointed out that the number of output neurons can be higher than one, as much as the that of the hidden layers.

c) **Bridge network**: to build this architecture, the interconnection between neurons of different layers does not have follow a specific scheme, gaining the possibility to connect different layers in multiple ways. In the figure below a graphical representation of a bridge network is given for reference:

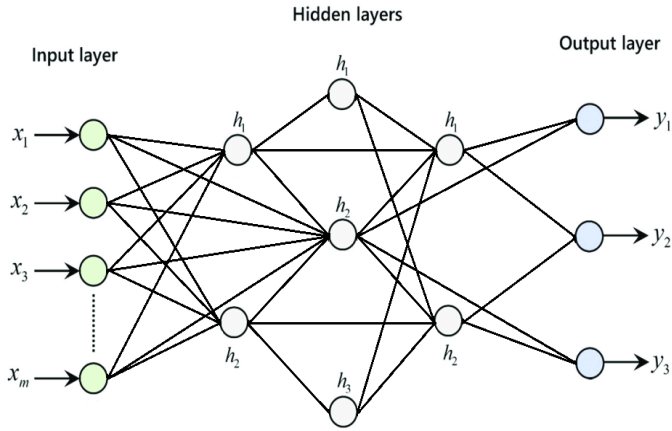

Figure 14: Interconnection scheme for a bridge network

The network in the figure is just one of the possible topologies that can be achieved in the framework of bridge networks. Indeed, alternative interconnection schemes and number of the neurons could be considered for building other configurations. There are no limits to the possible configurations of the network with this topology starting every time with the same number of hidden layers, output layers and neurons. The bridge strategy slightly simplifies the network with respect to a cascade configuration, reducing the number of weights. On the other hand, it could be difficult to determine an automated procedure that build the interconnections, resulting in a task to be performed by hand. The reduction of complexity from a cascade network reflects in a reduced training time. As said for the cascade network, the bridge network can solve problems that are solved by feedforward network of analogous architecture, while the opposite is not granted to hold.

d) **Radial Basis Function network**: this kind of network is made by one hidden layer only. It is named *radial basis function* network because the activation function inside the neurons in the hidden layer is, in most of the cases, a Gaussian function. The value of this particular type of function changes in relation to the distance from a central point.
This topology is easy in design and can solve very complex problem. Furthermore, the RBF network work better than other architecture if the input is affected by noise [29][35][36]. There is also the possibility to introduce the position of the centres inside the set of variables that have to be tuned during the training phase of the neural network. However, the increase of the parameters inside the training procedure could increase the time consumption.

Most of the problems can be solved with a neural network built with two hidden layers [2][6]-[8]. Generally, the neurons are equally distributed in every hidden layer, or are positioned in a decrescent way, where the first hidden layer has more neurons than the last one. If the ANN is too big, the network can work within a limited region of the space or with only a few sets, giving wrong results for other possible inputs. This is due to the generalization property of the artificial neural network that varies depending on the number of neurons and interconnection. In fact, as the number increases, the generalization decreases and vice versa. However, a minimum number can be found for both neurons and weight that allows the ANN to fulfil its task with an acceptable level of approximation.

The generalization property of an ANN depends also on the number of internal parameters linked to the number of neurons and on the architecture [34]. In fact, the right number of parameters is the one that balances the under-fitting and the over-fitting phenomena. In the figure below three different networks approximations are shown, where the value M is the number of hidden units (neurons per layer) inside the network:

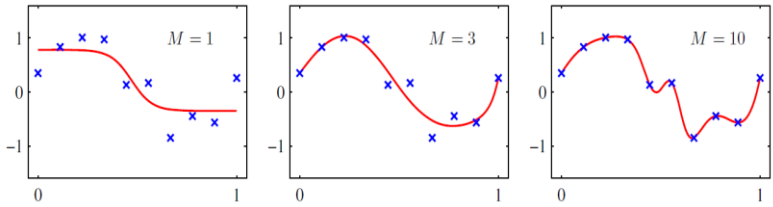

Figure 15: Two-layer fitting network with different hidden units [34]

It can be observed that as the number of parameters increases the under-fitting phenomenon reduces in magnitude, increasing the generality of the ANN.

4.1.4 Activation functions

Each neuron embeds an activation function that elaborates the input information to generate an output, as shown in Figure 11. A vast amount of function can be considered for this scope. These functions are scalar-to-scalar and are named "*threshold* functions" or "*transfer* functions".

The most popular activation functions are "*Uni-polar sigmoid*", "*Bi-polar sigmoid*", "*Tanh*", "*Conic Section*", "*Radial Basis Function*" (RBF), and "*Rectified Linear Unit function*" (ReLU). Other types of functions such as the "*identity* function", the "*step* function" or the "*binary step* function" could also be used but are not reported here for sake of conciseness. A brief description of the most popular alternatives is provided below [28][38][39]:

a) **Uni-Polar Sigmoid function**: this function maps the interval $(-\infty, \infty)$ onto $[0,1]$. This kind of function is especially advantageous to be used in neural network trained by back-propagation algorithms because it can minimize the computational resources needed to compute the gradients. The function is expressed in equation:

$$g(x) = \frac{1}{1+e^{-x}} \tag{4.1}$$

As a reference the activation function shape is reported in the figure below:

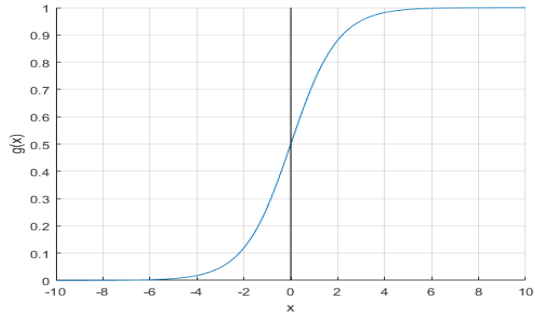

Figure 16: Uni-Polar Sigmoid function

b) **Bi-Polar Sigmoid function**: this function is similar to the Uni-Polar Sigmoid function, the only difference is in the codomain that is in the range of $[-1,1]$. The mathematical formulation of this kind of function is expressed in the following equation:

$$g(x) = \frac{1-e^{-x}}{1+e^{-x}} \tag{4.2}$$

The Bi-Polar Sigmoid function behaviour is shown in Figure 17.

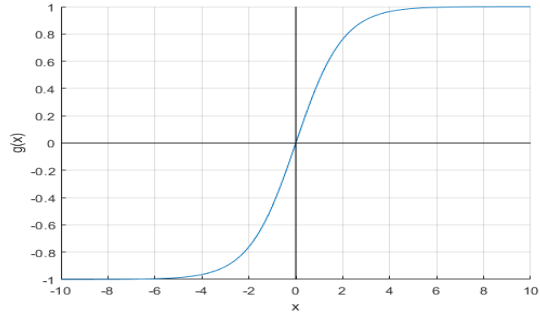

Figure 17: Bi-Polar Sigmoid function

c) **Hyperbolic Tangent function**: this function is defined as the ratio between the hyperbolic sine and the hyperbolic cosine functions as:

$$tanh(x) = \frac{\sinh(x)}{\cosh(x)} = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{4.3}$$

The Tanh is similar to the sigmoid function since its output ranges between -1 and 1, where the shape of the Tanh function is reported below:

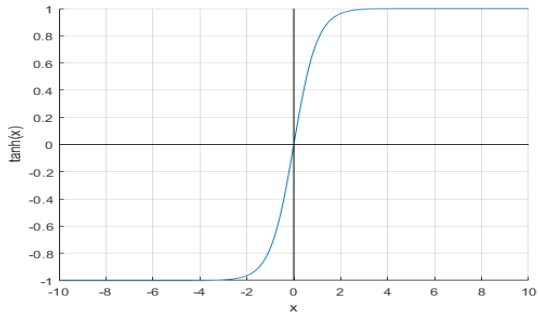

Figure 18: Hyperbolic Tangent function

d) **Conic Section function**: this kind of function is based on a section of a cone that is sliced by a plane. The parameters inside the function change the orientation of the plane that intersects the cone, modifying the shape of the cone section. In the following equation the formula of the conic section function is expressed:

$$f(x) = \sum_{i=1}^{N+1} (a_i - c_i)w_i - cos\omega(\|a_i - c_i\|) \quad (4.4)$$

where a_i is input coefficient, c_i is the centre, w_i is the weight linked with the neuron and ω is the opening angle which can be any value in the range of $[-\pi/2\,,\ \pi/2]$.
In the figure below a parabolic conic section is shown:

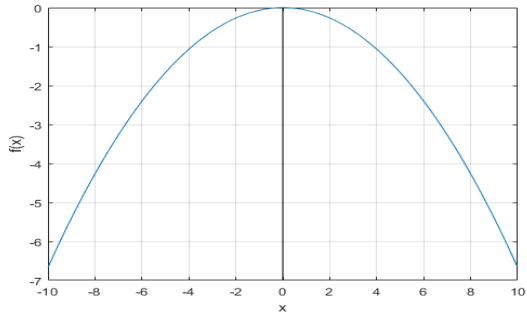

Figure 19: Conic Section function (parabola)

e) **Radial Basis function**: the RBF is based on Gaussian curves and takes parameters that determines the center (mean) value of the function used as desired value. This kind of activation function is a real-valued function whose value depends only on the distance from the origin, or alternatively on the distance from other point c (centre), as reported in equations:

$$g(x) = g(\|x\|) \quad (4.5)$$

$$g(x) = g(\|x - c\|) \quad (4.6)$$

Sums of radial basis function are typically used to construct function approximations of the form shown:

$$y(x) = \sum_{i=1}^{N} w_i \, g(\|x - c_i\|) \tag{4.7}$$

where the approximation function $y(x)$ is represented as a sum of N radial basis functions, each associated with a different centre c_i for each weight w_i coming from the neural network. As a reference in the following figure is reported a radial basis function with two centres collocated at $c_1 = 0.75$ and $c_2 = 3.25$ that generate the two unnormalized Gaussian radial basis function.

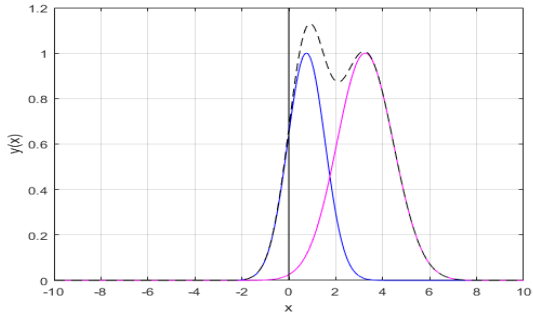

Figure 20: Radial basis function (with two centers collocated at c_1=0.75 and c_2=3.25)

The two curves are multiplied with different weights ($w_1 = 0.8$ and $w_2 = 1.2$). The dashed line is the function $y(x)$ of equation (4.7), referred to the summation of the two Gaussian curves.

f) **Rectified Linear Unit function**: the ReLU function is characterized by two different behaviours. For $x < 0$ the output of the function is zero for ReLU, while it is a linear function of the input in the case of parametrized ReLU functions. If $x \geq 0$ the function work as the identity function. These two types of function are expressed in equations (4.8 - 9), where the first is the ReLU function and the second is the parametrized one.

$$f(x) = \begin{cases} x & if\ x \geq 0 \\ 0 & if\ x < 0 \end{cases} \tag{4.8}$$

$$f(x) = \begin{cases} x & if\ x \geq 0 \\ ax & if\ x < 0 \end{cases} \tag{4.9}$$

where a could be assigned a-priori or it can be used as a parameter of the problem; in both the cases the coefficient a has to be positive, usually lower than 1. As a reference the two different kind of ReLU are shown in figure:

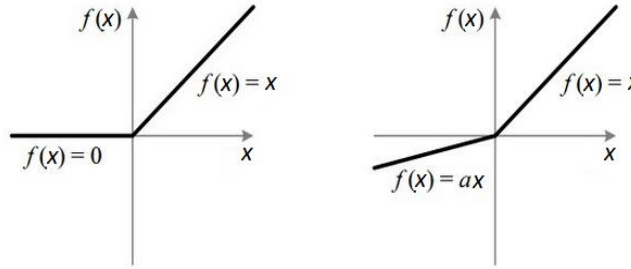

Figure 21: ReLU (left), and parameterized ReLU (right)

To avoid numerical problems, generally in the output layer the inputs of the neurons are passed through a linear activation function [6][29], i.e. identity function, or ReLU function. Therefore, the neuron maintains as output the weighted summation, or part of that.

4.2 ANN training

In this section, the general procedures of the training algorithm are reported. After presenting an overview, the following topics are briefly reviewed:

1. Overview.
2. Backpropagation algorithm (gBP).
3. Levenberg-Marquardt backpropagation algorithm (LMb).
4. Other kinds of training algorithms.

4.2.1 Overview

The artificial neural network has to be trained in order to perform its task. In fact, without a training procedure, the ANN is only an empty shell with nodes and interconnection between them without any useful usage. The training procedure is a tuning process in which all the weights and in general all the trainable parameters are updated several times. The weights are updated such as to minimize the error between the network output and the desired output. The training procedure is assumed to be completed when the error drops below a certain threshold.

There exists a huge number of different strategies to train a neural network. Some of this strategies are based on mathematical algorithms such as the gradient backpropagation (gBP) [34][40]-[42], or the Levenberg-Marquardt backpropagation (LMb) [34][43][44], other are based on hybrid algorithms that mix approach of the first order with others of the second order [34][45]; also some optimization algorithms as the PSO (particle swarm optimization) [26] or the GA (genetic algorithm) could be used to train the ANN. Every method differs from the others in terms of accuracy, time spent and reliability of the results.

All the training strategies have as aim the reduction of the error function that could be expressed in different ways. The most common error function is the mean square error [34][43]-[45]:

$$E = \frac{1}{2(np \cdot no)} \sum_{p=1}^{np} \sum_{m=1}^{no} e_{p,m}^2 \tag{4.10}$$

where p is the number of input-output sets, ranging from 1 to np, m is the number of output of the network that ranges from 1 to no, and $e_{p,m}$ is the error at output m defined as reported in equation (4.11).

$$e_{p,m} = o_{p,m} - d_{p,m} \tag{4.11}$$

where $d_{p,m}$ and $o_{p,m}$ are desired output and actual output respectively, at network output m for training set p.

In all training algorithms the same computations are repeated for one set of input-output at a time. Therefore, in order to simplify notations, the index p for sets will be omitted hereinafter, unless otherwise stated.

4.2.2 Backpropagation algorithm (gBP)

In the gradient backpropagation training strategy, the updated weights, that substitute the old ones in order to reduce the error shown in equation (4.10), are evaluated with the help of the gradient, as expressed in equation [34][41][44]:

$$w_{n+1} = w_n - \alpha g_n \tag{4.12}$$

where n is the index of iterations, w is the weight vector, α is the learning constant, and g_n is the gradient vector that is evaluated as:

$$g_{j,i} = \frac{\partial E}{\partial w_{j,i}} = y_{j,i}\delta_j \tag{4.13}$$

where j is the index of the neuron that ranges from 1 to nn, i is the index of neuron input that ranges from 1 to ni (this value can vary for different neurons), while δ_j is a parameters obtained from the output error, reported in the equation below:

$$\delta_j = s_j \sum_{m=1}^{no} F'_{m,j} e_m \tag{4.14}$$

where s_j is the slope of activation function f_j, and $F'_{m,j}$ is the derivative of a complex nonlinear function that connect neuron j with output m. The complexity of the nonlinear relation depends on how many neurons are between the neuron considered and the output. If the neuron j is at network output m, then $F'_{m,j} = 1$ [44].

The s_j term is determine as reported in equation:

$$s_j = \frac{\partial y_j}{\partial net_j} = \frac{\partial f_j(net_j)}{\partial net_j} \tag{4.15}$$

where the net_j is the summation between the weighted inputs and the bias error just before the value is passed through the activation function, and y_j is the output of the node, i.e. the value obtained by passing the net_j through the activation function.

4.2.3 Levenberg-Marquardt backpropagation algorithm (LMb)

In the Levenberg-Marquardt backpropagation algorithm, a Jacobian matrix is introduced to evaluate the updated weight vector, as shown in equation [34][44]:

$$w_{n+1} = w_n - (J_n^T J_n + \mu I)^{-1} g_n \tag{4.16}$$

where μ is the combination coefficient, I is the identity matrix, and J_n is the Jacobian matrix shown in Figure 22.

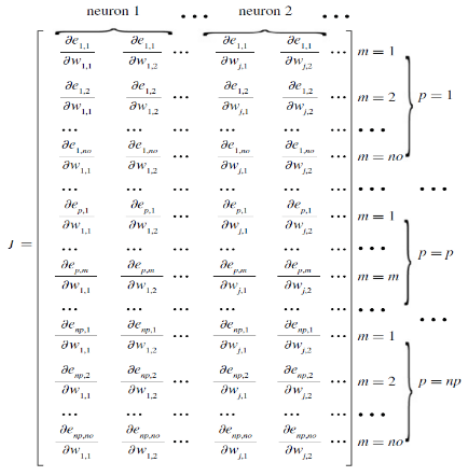

Figure 22: Structure of Jacobian matrix [44]

The number of columns of the Jacobian matrix is equal to the number of weights, while each row corresponds to a specified training set p and output m. The elements of the matrix are computed as reported:

$$\frac{\partial e_{p,m}}{\partial w_{j,i}} = y_{j,i}\delta_{m,j} = y_{j,i}s_j F'_{m,j} \tag{4.17}$$

where the parameter $\delta_{m,j}$ is the same evaluated in equation (4.14) with the only difference that this method is a second order algorithm and these parameters have to be calculated for each neuron j and each output m separately.

The gradient vector g_n in equation (4.16) can be obtained from partial results of the Jacobian calculations, as expressed below:

$$g_{j,i} = y_{j,i} \sum_{m=1}^{no} \delta_{m,j} e_m \tag{4.18}$$

The LMb algorithm uses a significantly higher number of parameters, as it can be deduced from the introduction of the Jacobian matrix that could be very large. The dimension of the matrix increases as the number of neurons, output and sets increase, inducing an increment in the time needed for the training [44]. On the other hand, the LMb can train artificial neural network for which the gBP algorithm has difficulty in converging. However, this method is suitable for small and medium size ANN.

In second order algorithms the starting point could be affect the final result [44], needing strategies focused on finding the initial values of the weights or heuristic approaches that remove this dependence. This latter strategy trains the ANN several times, without working directly on the initial values of the starting weights.

4.2.4 Other kinds of training algorithms

The hybrid algorithms mix in one strategy the time performance of the first order algorithms with the accuracy and the convergence capacity of second order algorithms. This kind of strategy needs a huge set of parameters to tune all the internal formulas in order to grant the convergence [45].

For what concerns the training procedures based on optimization algorithms such as PSO or GA the weights are the design variables. These parameters are obtained minimizing the objective function made by the error between the network output and the desired one [26]. In this case the design variables do not have any kind of upper or lower boundary that limit their values, without the needs of introducing some penalty terms inside the objective function.

4.3 Training and Testing sets

All the training procedures explained in section 4.2 need a training set containing a certain number of input-output pairs. The dimension of the set determines the accuracy of the final neural network. In fact, the more sets are available, the better performances are expected from the trained ANN.

The training sets are obtained with numerical analysis, i.e. FEM or Ritz, or with direct observations and measurements. The outputs of the training set represent

the desired values that the network has to approximate, tuning the internal weights.

After the training procedure is completed, the network must be tested to verify if there are possible lack of generalization, as introduced in section 4.1.3. This verification process could be done with other sets of inputs-outputs pairs, named testing set. This kind of sets are built without particular attention on their distribution inside the feasible design space, aiming only on the verification of the constrains by the lamination parameters. Therefore, the LHS strategy implemented in Chapter 3 can be substituted with a simple random determination of the values, checking if the LP sets obtained verify the constraints. The testing points are passed as inputs to the trained ANN, whose outputs are then compared to the exact ones and the generalization errors are computed.

The difference between the training error and the testing error behaviour is shown in Figure 23. The generalization error is the testing error, while the capacity is referred to the number of weights inside the network.

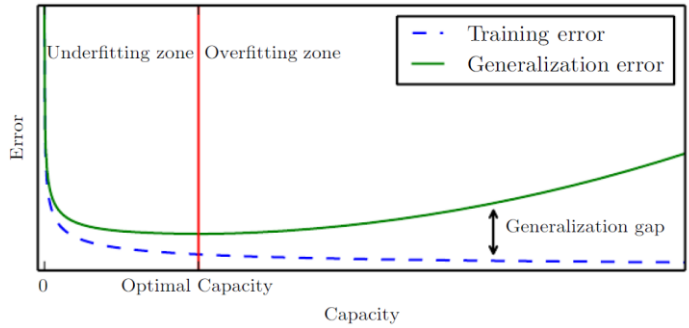

Figure 23: Training error and testing error behaviour as the number of parameters increases [34]

As explained in section 4.1.3, the under-fitting problem can be solved increasing the number of internal parameters. However, as shown in Figure 23, as the number of these parameters increases, the training error decreases, while the testing one increases. This behaviour is due to the fact that the network starts to over-fit the data used in the training process. Therefore, the ANN cannot be able to predict the

values of the points that are not used. In this case, it is said that the network "memorizes" the data, but not "learns" from them [34].

The "Optimal Capacity" reported in Figure 23 is referred to as the most efficient point, where the testing error reaches its minimum value. This condition is obtained with a precise number of neurons and weights. In order to achieve the optimal capacity situation, the number of training sets has to be 50 or even 100 times the number of weights present in the ANN [34].

4.4 Transfer-learning

This section focuses on a main question: is it possible to reduce the time spent to train two different neural networks, training only one ANN and using that as the starting point to tune the parameters of the other one? The answer to this question is to be found in using the transfer-learning strategy.

The transfer-learning is more efficient if it is used a reduced number of input-output sets for the training process. The advantage of using a reduced set consists in a smaller time consumptions for the training for the second ANN and also for the time spent to obtain the input-output pairs itself (if obtained with numerical iteration methods). The transfer learning works in two steps: the first step consists of training an ANN, and the second one consists in taking all the internal parameters of the first neural network and using them as the starting points for the training of the second ANN. As a reference, the scheme of the transfer learning procedure is reported in Figure 24.

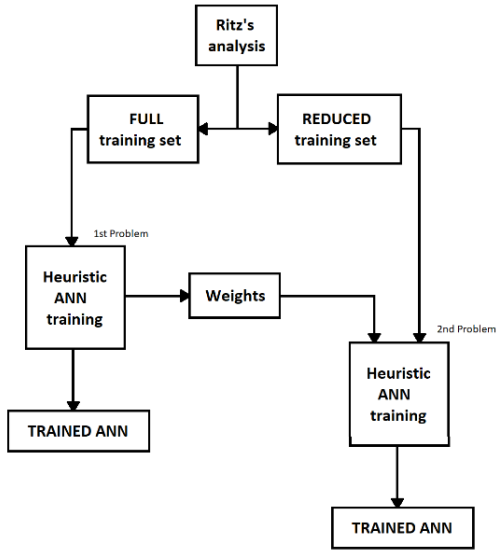

Figure 24: Procedure scheme for the transfer-learning strategy

One first point to be clarified regards the ability of transfer-learning to guarantee accurate prediction for both the problem considered. Note, in this framework the same network architecture should be considered for both the analysis. Therefore, the second problem could use an ANN that is not one of the most efficient for the resolution of the problem considered.

If the second problem network is trained with a “full” training sample set, the transfer-learning strategy could only reduce the training time. In fact, there are no differences between the classical training method and the transfer-learning method, except for the desirable proximity of the set of parameters to the convergence conditions. Instead, with a reduced set, also the time spent in the parts before the training can benefit with this kind of strategy.

4.5 ANN implementation in MATLAB®

Inside the software MATLAB® are already be implemented the code to build the ANN, and to train the network. An input files that contains all the information about the neural network and the training and testing input-output pairs has to be implemented.

The ANN can be built with the functions "*feedforwardnet*" or "*cascadeforwardnet*", where the input is a vector containing the numbers of neurons in each hidden layer. Furthermore, in the input file can be implemented a series of Boolean matrices that contains all the information about the interconnections between the various layers, modifying the classical scheme. Inside the input files it can be decided also if a bias has to be applied to a hidden layer or not, and in addition, which activation function is presented in each neuron.

The training algorithm must be selected introducing the relative string name inside the input file, i.e. "traingdx" for a classical backpropagation, "trainlm" for a LMb, and "traincgf" for a hybrid training algorithm.

Inside MATLAB®, however, the testing phase is not implemented, needing to write the codes for it. In order to accomplish this task, a function containing the trained ANN has to be generated. This can be done by hand or using the already implemented "genFunction". After that, a *for* loop is needed in order to evaluate all the testing errors for the input-output pairs: the input is introduced inside the function generated, and the output is compared with the desired one. Generally, the errors computed are the maximum and the average ones.

Both for the training and the testing errors evaluation, a *for* loop has to be coded. This can be also useful if a heuristic strategy is chosen to train the network, iterating the training process several times. In fact, these errors can be used inside an objective function to determine which one of the trained configurations is the best one.

For what concerns the transfer-learning implementation, one input file is needed which contains not only the input-output pairs for the first problem, but also for the second one. The training procedure and the definition of the network have to be put in sequence, starting with the training of the first ANN as implemented before and followed with the definition of the second one. At this point, the weights of the network should be extrapolated and introduced as starting points inside the

training algorithm of the second network, and the second network is trained. The second training process can be the same used for the training of the first ANN or can be even different from the previous one. It has to be underlined that the weights that connect the inputs with other neurons are not accepted by MATLAB® as starting points for the training. In fact, the only weights that can be transported from the first network to the training of the second one are the bias and the weights that connect the different neurons in the network.

Chapter 5

Particle Swarm Optimization (PSO)

In this chapter the optimization algorithm is exposed. After the definition of the problem with a surrogate model, the next part of the study is focused in showing how to reach the optimum value, i.e. maximum stiffness with minimum weight, maximum first buckling load or maximum first natural frequency.

Between all the possible strategies, the algorithm chosen is the Particle Swarm Optimization (PSO). The PSO is a stochastic process that possesses a natural-based algorithm, whose purpose is to mimic the bee's behaviour in mapping and finding the best spot where to collect food.

The artificial neural network is transformed into the objective functions, whose values are the ones that the PSO algorithm tries to maximize.

The Particle Swarm Optimization algorithm used in this work is the one implemented in MATLAB®, where some modifications are implemented in order to allow a verification of complex constraints. In particular, the constraints considered are referred to the lamination parameters that are implemented as design variables.

5.1 PSO Algorithm

The Particle Swarm Optimization algorithm is composed by a swarm of particles (bees) that represents various combination of deign variables with the relative solution value. The swarm of particles moves in the design space, searching for the maximum value, changing the velocity and the direction of the movement of every particle in order to find the optimum.

The use of PSO is well established in the literature [26][46][50]. For composite structural problems, the study reported in literature [46][50] demonstrated the PSO efficiency, showing that the performance obtained with this strategy can be better

respect to the ones obtained with other kinds of evolutionary optimization algorithms.

The PSO algorithm has its advantages and disadvantages. On one hand the time needed to reach the maximum is very small, on the other hand the convergence to the global maximum is not granted. To avoid being trapped in local optima, the internal parameters need to change during the process in order to modify the kind of search, i.e. from the global search to the local one. Therefore, some internal parameters have to be tuned [11].

The particles in the algorithm are placed in a multidimensional space that correspond to the feasible design space of the problem. Each particle can move in this hypervolume where the coordinates that describe its position are the values associated with each design variables. Every particle is associated also with the value that has to be optimized. This is done in to ensure that the swarm converge to the best spot (the maximum searched value).

The distribution of the swarm inside the design space does not affect the convergence property of the optimization algorithm. However, the dimension of the starting swarm could change the performance of the optimization process [46]. A larger number of particles allows to explore more space with a smaller number of iterations, but on the other hand this increase the time spent for the evaluation of the function value.

The ideal number of the swarm of particles is between 2n and 10n where n is the number of input variables [11][46]. If the number of particles of the swarm is less than 2n, the optimization process does not work and the convergence is not obtained, because particles move almost randomly along the design space. On the other hand, if the number of particles exceed ten times the number of the input design variables, the number of function evaluation becomes larger with no additional benefits on the ability of the method to reach the optimum.

The scheme of a basic PSO algorithm is shown in Figure 25.

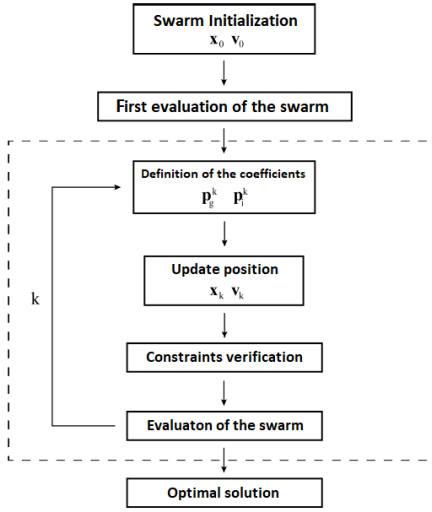

Figure 25: PSO algorithm scheme

The PSO algorithm consists of an iterative procedure, where at each iteration the position of each particle in the swarm is updated. The equation describing the change of position of a particle reads [11][27][46]:

$$x_{k+1}^i = x_k^i + v_{k+1}^i \Delta t \tag{5.1}$$

where x_{k+1}^i is the position of the particle i at iteration $k+1$ and v_{k+1}^i is the corresponding velocity vector; Δt is a time step that generally is considered unitary. There exist different ways to update the velocity vector referred to each particle, depending on the PSO algorithm under consideration. A commonly used scheme is expressed in equation:

$$v_{k+1}^i = \omega v_k^i + c_1 r_1 \frac{\left(p^i - x_k^i\right)}{\Delta t} + c_2 r_2 \frac{(p_k^g - x_k^i)}{\Delta t} \tag{5.2}$$

where r_1 and r_2 are independent random numbers between 0 and 1, p^i is the best position found by particle i and p_k^g is the best position in the swarm until iteration k. The inertia of the particle ω and the two "trust" parameters c_1 and c_2 could change during the iterative procedure to try to find the global maximum and avoiding that the optimization algorithm will be trapped in a local optimum [11][46]. Both the inertia of the particles and the trust parameters are non-dimensional values.

The inertia controls the exploration properties of the algorithm. When the inertia has large values a more global behaviour is enhanced, while with smaller values the analysis is more local. In the first part of the process many regions are explored, while during the final phase the particles move less and less around their positions. Therefore, the more promising spots are found in the first phase, while in the second one, the particles adjust the final optimal configuration without the risk to jump away from that spot [11][46].

The trust parameters indicate how much confidence the current particle has in itself (c_1), and how much confidence it has in the swarm (c_2). Furthermore, also the trust parameters can change during the iteration process in order to optimize the convergence [11][46].

After all the positions are updated, any constraints on the design variables are checked. Different strategies can be used to consider possible violation of the constraints, i.e. modifying the velocity vector [11] or changing directly the objective function value associated with the particle [21].

At each iteration, the value associated to a particle is compared with the others, and the best one of the swarm is saved. The last saved value is the optimal solution when the stop criterion ends the iteration process. Therefore, the design variable combination associated to that particle is the optimal combination of design parameters.

There exist different ways to stop the optimization procedure, i.e. maximum number of iterations, maximum time consumption allowed, the best value is below the limit imposed, or the best value does not change within a "stall maximum" amount of iterations.

The maximum number of iterations is the most adopted stopping criterion for the PSO algorithm. Furthermore, this criterion can be placed side by side with the stall maximum iteration stop criterion. However, it may happen that the objective

function stays stationary for a certain number of iterations and then starts to change its value again, reason why the former criterion is often preferred.

The scheme of the algorithm is reported below:

1) Start with an initial set of particles, randomly distributed throughout the feasible design space.
2) Choose the inertia of the particles, depending on the number of the current iteration, to modify the type of search (more local or more global).
3) Calculate the velocity vector for each particle in the swarm.
4) Update the position of each particles using previous position and the velocity calculated in point 3.
5) Store the best position of the swarm and the best position of each particle obtained so far, for the iteration that follows.
6) Go to step 2 and repeat until the stop criteria end the process.

5.2 Artificial neural network in the optimization process

Within the present framework, the objective function is determined using an artificial neural network. The ANN can be converted into a function where the inputs are the coordinates of the particles and the output is the value of the objective function. In this way the optimization algorithm uses the ANN inside the iteration process. A scheme of the entire optimization process is shown in Figure 26.

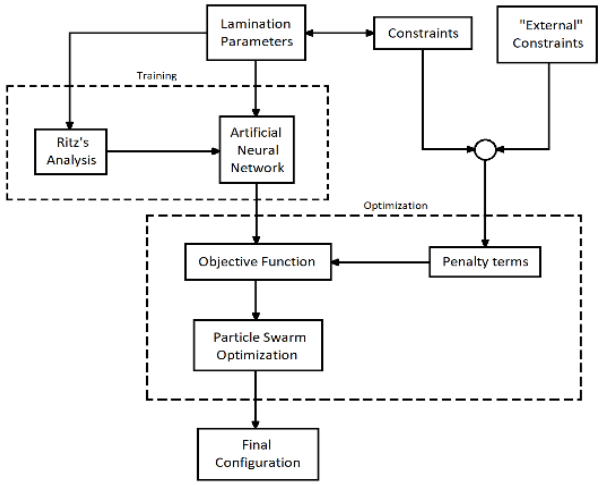

Figure 26: Optimization process scheme, including the numerical analysis done with a Ritz's method and the training of the ANN

The optimization algorithm treats the ANN as a Blackbox, providing inputs to the net which in turns generates the corresponding output. The optimization algorithm operates on the output provided by the ANN, applying penalty terms enforce the constraints. The advantage of using neural networks consists of an almost immediate evaluation of the objective function.

Errors can be affecting the optimization process due to the presence of the ANN. In fact, the percentual error on the testing sets, explained in Chapter 4, remains also in the optimization process.

5.3 Constraints application

The PSO algorithm needs strategies to check any constraints that can be applied to the design variables. Several strategies are available for preventing the particle to fall outside the feasible domain. Therefore, the updated positions of the particles

are ensured to be always inside the design domain. The constraints applied to the problem could be physical, parametrical or made by design choice.

The most popular strategies are *"absorption"*, *"reflection"*, and *"damping"*, reported as a reference in the figure below [11]:

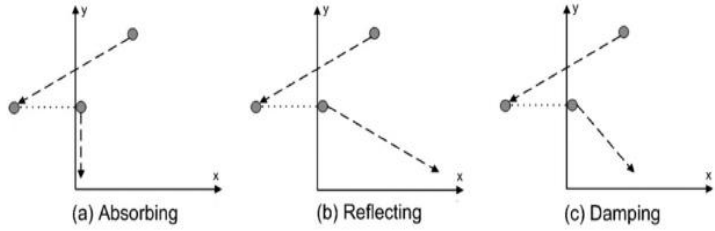

Figure 27: The three most popular boundary condition method in PSO

These three methods with their relative implementation are listed below:

a) **Absorption:** this method assigns to the degrees of freedom of the particles that violate the constraints the limit bound value. This imply that the particle is brought on the boundary limit of the violated degrees of freedom ad absorbing their velocity, reducing those to zero. The absorption method fits better the problem that try to find solution near the boundary limits because the particles that try to escape the feasible domain trend to remain stacked on the boundary limit line, analysing better those regions rather than the internal ones. The algorithm could be expressed in a mathematical way as reported in equation:

$$\begin{aligned} if\ (x^{ni} > x^{n}_{\max}) \quad &\rightarrow \quad x^{ni} = x^{n}_{\max} \\ if\ (x^{ni} < x^{n}_{\min}) \quad &\rightarrow \quad x^{ni} = x^{n}_{\min} \end{aligned} \tag{5.3}$$

b) **Reflection:** this method is similar to the absorption one, in fact the particles is bought on the boundary that was about to be violated. The main difference is on the velocity vector that the particles has after the repositioning; the velocity is not absorbed but reflected. The components of the velocity vector referred to the violated degrees of freedom are not set to zero, but their magnitude is maintained while their sign is reversed. The reflection method works like if the particles was elastically bounced back from the boundary limit expressed as:

$$\begin{aligned} if\ (x^{ni} > x^{n}_{\max}) \quad &\rightarrow \quad x^{ni} = x^{n}_{\max} - cv^{ni} \\ if\ (x^{ni} < x^{n}_{\min}) \quad &\rightarrow \quad x^{ni} = x^{n}_{\min} - cv^{ni} \end{aligned} \tag{5.4}$$

where *c* is a scalar number that is unitary in the classical case (perfect elastic bouncing) that in most of the cases gives problem to the convergence because the particles that are replaced tend keep violating the constraints. For this reason, the value of the scalar *c* tends to assume different values accordingly to the problem.

c) **Damping:** this method conceptually coincides with the method of reflection with the adding of a random damping between 0 and 1 to the "bouncing" velocity as shown in the equation below:

$$\begin{aligned} if\ (x^{ni} > x^{n}_{\max}) \quad &\rightarrow \quad x^{ni} = x^{n}_{\max} - (rand)cv^{ni} \\ if\ (x^{ni} < x^{n}_{\min}) \quad &\rightarrow \quad x^{ni} = x^{n}_{\min} - (rand)cv^{ni} \end{aligned} \tag{5.5}$$

In this method a randomness component is introduced by the *rand* term. Both the *reflection* and the *damping* methods fits better problem that try to reach optimal point away from the boundary limits, analysing more thoroughly the internal regions of the feasible design space.

Another approach to enforce constraints consists in introducing a penalty term in the objective function [21]. Furthermore, this strategy allows to introduce a large set of constraints, even complex or nonlinear ones.

The easiest way to introduce a penalty term is to use a flag that becomes true (= 1) if the constraints are violated, and false (= 0) if they are not. A penalty value is multiplied by this flag terms, increasing or decreasing the value of the objective function. The mathematical formulation of this concept is expressed as:

$$F(\boldsymbol{x}, \lambda) = f(\boldsymbol{x}) \pm M\lambda \tag{5.6}$$

where $F(\boldsymbol{x}, \lambda)$ is the penalized objective function that depends on the vector $\boldsymbol{x}$ of the project variables and λ that is the flag: $\lambda = 0$ if the constrains are verified by $\boldsymbol{x}$ and $\lambda = 1$ if $\boldsymbol{x}$ violates the constraints. The value of M is large enough to bring the value of the objective function $f(\boldsymbol{x})$ out of the set of possible results.

Another option to introduce the penalty term inside the objective function is to consider not a flag that multiplies a large number M, but an error computed from the constraints itself. This allows also to analyse the outer space near the boundaries, in order to better understand the behaviour of the problem itself. The mathematical formulation of this strategy is expressed in the equation below:

$$F(\boldsymbol{x}, e) = f(\boldsymbol{x}) \pm Me \tag{5.7}$$

where e is the maximum error obtained in the checking phase of the constraints.

The equations (5.6) and (5.7) can be combined when several constraints are considered. Furthermore, these applied constraints could be of different nature, e.g. errors and Booleans. For the penalty term reported in equations (5.6) and (5.7), the value M can be considered proportional to the value obtained from the objective function:

$$M(\boldsymbol{x}) = a \cdot |f(\boldsymbol{x})| \tag{5.8}$$

where a is a scalar that have to be chosen in order to modify significantly the value of the objective function. Therefore, the value a has to have an appropriate order of magnitude: if it is too small the particle that violates the constraints can be considered anyway; on the other hand is meaningless that it is too large, enlarging the objective function value beyond what is necessary.

The implementation of the constraints with the use of a penalty terms could introduce errors inside the optimization process. Considering this point, the total error present in the procedure is a combination of the one produced by the constraints and the one intrinsic in the ANN.

5.4 PSO implementation in MATLAB®

Inside MATLAB® is already implemented the basic code for the particle swarm optimization ("*particleswarm"* function). However, in MATLAB® the method implemented to check if the constraints are verified is the absorption one. Therefore, the implementation of penalty terms inside the objective function is needed.

The ANN is used as objective function inside the PSO algorithm, generating a function as done in Chapter 4 to evaluate the testing errors. If the constraints present are only upper and lower boundaries, the objective function consists directly of the function generated with the ANN. However, if this is not the case, after the ANN a checking phase and an evaluation of the errors need to be introduced.

The value outcoming from the ANN and the penalty terms generated by the checking of the constraints are summed together and the final value is the one associated to the particle during the iteration process.

The inertia of the particles, the trust parameters (c_1 and c_2), the swarm size, and the maximum number of iteration (stop criterion) are parameters that can be modified using the string "*options*".

The range given by default in MATLAB® for the inertia parameter is between 0.1 and 1.1. Therefore, it is proved in literature that this interval can grant the convergence [11]. For what concerns the trust parameters, in literature is

demonstrated [11][46] that if their value change during the iteration, the convergence is faster. However, in MATLAB® the trust parameters do not change during the iteration process and by default they start with value of 1.49.

The number of particles and their coordinates (design variables values) for the initial swarm can be defined. It has to be noticed that if particular constraints exist, also the particles of the initial swarm have to verify them.

The PSO algorithm implemented in MATLAB® is coded to find only the minimum optimal value. However, if the maximum value is the desired ones, the dual problem has to be considered. In fact, the dual problem transforms the optimization search from the minimum value to the maximum one or vice versa, introducing some transformation on the variables and unknowns involved.

5.5 Validation of the PSO process implemented

In this section the validation of the PSO algorithm used in the optimization process is presented. This validation checks if the convergence is obtained even if the objective function is built with an ANN and penalty terms are introduced.

To validate the algorithm implemented, a specific problem is considered. The mechanical properties of a simply supported rectangular plate with dimension of 1200x1000 mm and made by Carbon Fiber are reported in the table below:

	Values
E_{11}	163000 MPa
E_{22}	6800 MPa
G_{12}	3400 MPa
ν_{12}	0.28
ρ	1.35E-9 t/mm^3

Table 6: Composite material engineering properties

The Plate considered is made by 3 plies with a constant distribution of the orientation of the angles in each layer, where the thickness is fixed to be equal to 0.127 mm for each ply. No particular assumptions, like symmetry or balancing, are imposed to the plate.

The optimal value for the free vibration problem is evaluated with two distinct methods: the PSO algorithm with ANN as objective function and penalty terms, and a numerical iteration considering all the possible different orientation angles combination to have a reference benchmark.

The two methods use different design variables. In fact, the PSO optimum value is reached using lamination parameters, while the one obtained with the numerical iteration uses the orientation angles of the fibres. In this latter case, the angle of one ply at each step is increased by 5° because the difference in properties can be considered constant in that interval [22]. Therefore, 46656 different laminates are considered and are introduced in a Ritz's analysis to evaluate the respective frequencies.

The ANN used to validate the results is composed by 2 hidden layers with 12 neurons each with a feedforward architecture. It is trained with a heuristic method, composed by 20 iteration of LMb training algorithm.

The training points are obtained with the same procedures expressed in Chapter 3, and the training set used is composed by 796 points. In particular, the number of terms needed to fully describe the laminate is 12 (one set only).

For what concerns the PSO, the size of the swarm is about 2n, where n is the dimension of the design variable set. The trust parameters c_1 and c_2 are 1.86 and 1.61, respectively. Finally, the stop criteria are imposed to be 5000 iteration and a maximum stall iteration of about 20% of the maximum number of iterations.

The values of the first natural frequencies obtained with the two different method are reported in table:

LP + ANN + PSO [Hz]	ϑ + Numerical Iteration [Hz]
4.4119	4.4135

Table 7: Optimal results obtained from the PSO algorithm and the numerical iteration

It can be observed that the difference between the two values in Table 7 is in the order of 0.04%. However, the particle swarm algorithm that uses the ANN as objective function, is affected by errors. The first natural frequency is obtained from a Ritz's analysis starting from the lamination parameters obtained by the PSO. Therefore, the actual value is 4.3947 Hz, generating in this case a percentage error in the order of 0.43%.

The optimization process that use the PSO algorithm with ANN as objective function and lamination parameters as design variables that introduce complex constraints, can achieve good approximation of the actual optimal value. Furthermore, less time is spent in the PSO optimization process respect to the one spent in the numerical iteration. The time needed to find the optimal value with a numerical iteration is about 23 hours, while the PSO spent is less than 3 hours. This means that in half of the time, the optimal solution is found with a small error.

Chapter 6

Post-processing

In this chapter is explained and validated the post-processing procedure to transform lamination parameters into orientation angles. Due to the nature of the lamination parameters, the conversion is performed numerically. In fact, different configurations of angles per ply could be obtained, depending on the number of layers and their thickness. Furthermore, the number of possibilities increase if the laminate is made by variable stiffness composite layers.

Different strategies can be implemented to solve the final transformation problem. The PSO algorithm is chosen to perform the conversion. However, the weights inside the objective function has to be linked with the physics of the structure, like the strain energy, in order to build a laminate as similar as possible to the starting one. The transformation executed in the post-processing part is mandatory if the laminate would be manufactured.

Constraints can be implemented in the conversion process in order to obtain stacking sequences that follow some design requirements.

6.1 From lamination parameters to orientation angles

The transformation of the angles in lamination parameters is unique, while opposite is not true. A numerical method is required to successfully convert the lamination parameters into the set of orientation angles. In literature different strategies have been developed for this scope. Friswell et al. [21] used an optimization algorithm, while Setoodeh et al. [4] introduced a stream function. In this work the optimization algorithm is implemented to convert the lamination parameters.

The PSO algorithm is used in order to convert the lamination parameters in orientation angles. The design variables are the angles that are converted in lamination parameters and then transformed again to compare their values with

the target ones [21]. As a reference, a scheme of the optimization process to determine the configuration of the orientation angles is reported below:

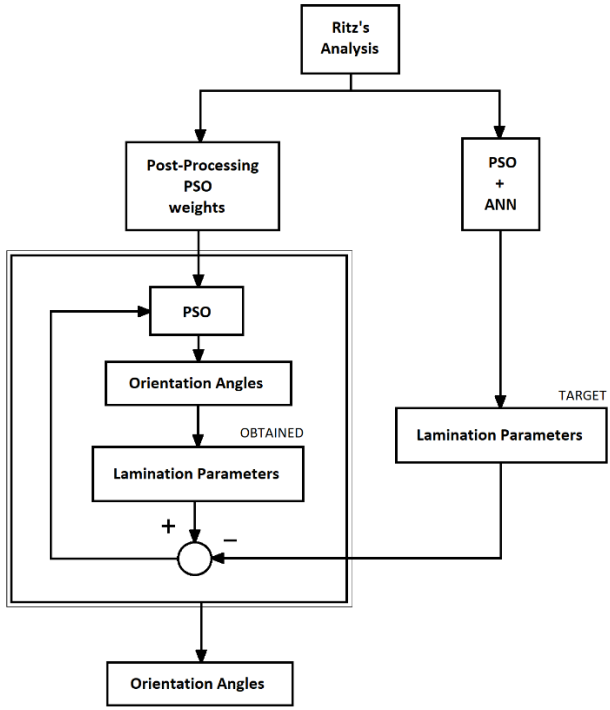

Figure 28: Scheme of the post-processing procedure

The objective function is composed by all the quadratic weighted error differences. Furthermore, there is the possibility to introduce a various set of constraints in the conversion procedure. However, in this work the simplest post-processing transformation is implemented, without the introduction of penalty terms.

The objective function proposed by Friswell et al. [21] is composed by linear error differences. However, if there are errors in the process that generates the optimal

set of lamination parameters, a quadratic error should be chosen in order to reduce possible errors that will be introduced by the conversion. The objective function is of the form reported in equation:

$$F(\boldsymbol{\xi}) = \frac{1}{12 \cdot np}[\sum w_a(\xi^A - \xi_t^A)^2 + \sum w_b(\xi^B - \xi_t^B)^2 + \sum w_d(\xi^D - \xi_t^D)^2] \qquad (6.1)$$

where ξ_t^k is the target lamination parameter associated to the *k* laminate constitutive matrix, while w_a, w_b, and w_d are the weights associated to the different components of the lamination parameters. Each term of the objective function is a quadratic error, evaluated component per component, subdividing the three different behaviour of the laminate structure. The np value is referred to the number of sets that are used to describe the laminate. Therefore, for a constant stiffness laminate $np = 1$, while for a variable stiffness plate np could increase significantly in function of the complexity of the fibres in-plane variation pattern.

The outline of the post-processing conversion is as follows:

1) Defining the target lamination parameters.
2) Evaluating the internal parameters of the PSO (objective function weights).
3) Generating a starting swarm of orientation angles.
4) Updating of position and velocity of the particles, as expressed in section 5.1.
5) Converting the angles in lamination parameters.
6) Evaluating the objective function value through the errors.
7) Repeat from point 4 until stop criteria end the process.

When the process is ended by the stop criterion, the set of project variables, that gave the minimum value of $F(\boldsymbol{\xi})$, expresses the angle distribution in each ply. It has to be observed that in equation (6.1) nor constraints or particular restriction are considered. In fact, as introduced before, inside the objective function $F(\boldsymbol{\xi})$ could be introduced penalty terms, i.e. the maximum difference between the angles in two consecutive plies or avoiding certain angle configuration in certain plies.

6.2 Weights of the objective function

The weights reported in equation (6.1) give more importance to certain type of lamination parameters components that have a larger influence on the structural behaviours. In order to assign the right level of importance, the weights cannot be assigned randomly but have to be referred to some sort of physical quantities.

The strain energy is chosen to extract the weights because it rates the contribution of the in-plane, coupling, and out-of-plane laminate constitutive matrices influence on the structure [22][51]. If the energy associated with a laminate constitutive matrix is null, the relative weight will be zero, neglecting all the errors on the associated lamination parameters terms. Therefore, these weights have to be modified to consider the conversion errors of the associated terms, otherwise the obtained laminate will be different from the starting one.

The strain energy is defined as reported in the following equation [22][51]:

$$U = \int_V \frac{1}{2} \begin{Bmatrix} \varepsilon_x \\ \varepsilon_y \\ \gamma_{xy} \end{Bmatrix}^T \begin{Bmatrix} \sigma_x \\ \sigma_y \\ \tau_{xy} \end{Bmatrix} dV = \int_\zeta \frac{1}{2} \begin{Bmatrix} \varepsilon^0 \\ k \end{Bmatrix}^T \begin{Bmatrix} N \\ M \end{Bmatrix} d\zeta \tag{6.2}$$

where ζ is the surface area of the midplane of the laminate, U is the strain energy, and ε^0 is the mid-plane strain vector, while k is the plate curvature vector; N and M are the membrane loads vector and the out-of-plane moments vector resultants, respectively. From equation (6.2) the strain energy can be decomposed in terms referred to the relative laminate constitutive matrix. The decomposition is reported [51]:

$$U = \int_\zeta \frac{1}{2} \begin{Bmatrix} \varepsilon^0 \\ k \end{Bmatrix}^T \begin{bmatrix} A & B \\ B & D \end{bmatrix} \begin{Bmatrix} \varepsilon^0 \\ k \end{Bmatrix} d\zeta = \int_\zeta \frac{1}{2} (\varepsilon^0 A \varepsilon^0 + \varepsilon^0 B k + k B \varepsilon^0 + k D k)\, d\zeta =$$

$$= U_A + U_B + U_D \tag{6.3}$$

where each term U_k, with $k = A, B, D$, are expressed below:

$$\begin{aligned} U_A &= \int_\zeta \frac{1}{2} \varepsilon^0 A \varepsilon^0 \, d\zeta \\ U_B &= \int_\zeta \frac{1}{2} (\varepsilon^0 B k + k B \varepsilon^0) \, d\zeta \\ U_D &= \int_\zeta \frac{1}{2} k D k \, d\zeta \end{aligned} \tag{6.4}$$

where U_A, U_B, and U_D are the three components referred to the different laminate constitutive matrices A, B, and D, respectively.

In practice, the integrals involved in equation (6.4) are computed numerically. From the three terms of the strain energy, the relative weights are evaluated. All the terms that are different from zero are reduced to be less than 10. For what concerns all the possible terms of the energy that are zero, the value of the relative weight is evaluated from the maximum value of the strain energy of the other terms. Therefore, they are scaled until the weight reaches a value between 0.5 and 1 in order to reduce their influence inside the optimization procedure, but still considering the error produced by those terms.

6.3 Post-processing implementation in MATLAB®

The post-processing process is implemented in MATLAB® in order to obtain the orientation angle configuration and the error between the target lamination parameters and the ones obtained converting the angle set. Therefore, a function that converts the angles into lamination parameters is needed to check the final errors. Furthermore, the PSO algorithm use the orientation angle as design variables, thus the conversion to lamination parameters is needed also inside the objective function. However, the "*particleswarm*" function admits as inputs only the objective function and "*options*". This implies that some data has to be written by hand inside the objective function code, i.e. the lamination parameters set, the weights obtained from the strain energy, the number of layers, and the total thickness of the plate.

If symmetric laminates are considered, the code can be only use as design variable half of the total number of angles. For what concerns other kind of laminate structure, such as balanced ones, the implementation could be more complicated.

6.4 Validation of the post-processing procedure

The post-processing has to be validated. Therefore, some laminates with fixed orientation angles are considered, converting these angles in lamination parameters and then converted again in angles with the PSO algorithm. Furthermore, to evaluate the capability of the algorithm implemented, constant stiffness laminate and variables stiffness laminate are considered in the validation phase.

All the laminates considered are symmetric and balanced. For this procedure, the trust parameters are fixed both to the value of 2.05 [11], while the weights of the objective function are computed from the strain energy for each laminate. The swarm is composed by 500n particles and the maximum iterations are 3000, where 10% of this is the maximum "stall iterations".

The laminates considered and the maximum error on the lamination parameters obtained in the validation process are reported in the following table:

Laminate stacking sequence	LP maximum error
$[0/90]_s$	0
$[\pm 45/\mp 60/\pm 30]_s$	4.81E-5
$[\pm 45.75/\pm 15.62/\mp 78.11]_s$	1.32E-5
$[(0-90)_2]_s$	0
$[\pm 45-0/0-\pm 45/\pm 45-0]_s$	5.83E-4
$[\pm 73.25-90/\pm 32.12-\pm 53.82]_s$	1.44E-4

Table 8: Validation laminates with the relative maximum errors on the lamination parameters due to the conversion

For the VSP laminates, the $[\vartheta_1 - \vartheta_2]$ notation is used, where ϑ_1 is the angle in the center of the plate, while ϑ_2 is the one on the edge. From the errors reported in Table 8, the error on the lamination parameters are negligible. Considering this observation, the method implemented it is reliable and can built the laminate starting from the lamination parameters. However, another validation test is studied.

The laminates considered in the previous validation test are used also for this study. However, for this analysis, the number of layers of the laminate does not match the starting one. In fact, in one case the number of plies is halved, while in the second one the number is doubled.

This analysis focus on the performance of the implemented code to find combination of angles, where the starting configuration cannot be obtained. The laminates considered for this study are reported in the table below:

Laminate stacking sequence	LP maximum error (half plies)	LP maximum error (double plies)
$[0/90]_s$	1	6.54E-5
$[\pm 45/\mp 60/\pm 30]_s$	0.3471	5.17E-4
$[\pm 45.75/\pm 15.62/\mp 78.11]_s$	0.2953	2.07E-4
$[(0-90)_2]_s$	0	0
$[\pm 45-0/0-\pm 45/\pm 45-0]_s$	0.3542	0.0045
$[\pm 73.25-90/\pm 32.12-\pm 53.82]_s$	0.3953	0.0030

Table 9: Validation laminates with the relative maximum errors on the lamination parameters due to the conversion for the case with half plies and the one with twice plies

From the data reported in Table 9 one can observe that the maximum error increase significantly when the number of plies is less than the original one. For what concern the case with more plies, the maximum error increases but remains negligible. Only the first VSP laminate considered has not changes in the error level, because the same lamination parameters set can be obtained repeating the top two layers an even number of times. These observations have to be considered in future problems, implying that large errors are principally related with a not sufficient number of layers.

Chapter 7

Numerical simulations

In this chapter all the numerical analysis based on two example problems are reported. The example problems considered are the free vibration and buckling one. Therefore, the first natural frequency and the first buckling load are the values to be maximized.

All the tuning process of the various parameters involved in the problems are reported, i.e. number of lamination parameters sets, weights of the ANN, inertia of the particle of the PSO, and so on. Furthermore, some assumptions and simplifications done during the analysis are reported in order to fully describe the work.

7.1 Assumptions and data for the example problems

The example problems consist in finding the maximum first natural frequency and the first buckling load for a simply supported rectangular variable stiffness plate.

In this work the simplest VSP configuration is considered, i.e. linear variation of the orientation angles through the plate. The orientation angles of the fibres in the center and on one edge are selected to fully describe the fibre pattern.

As a reference, the variation of the fibres inside each ply and the constraints applied to the edges are reported in Figure 29.

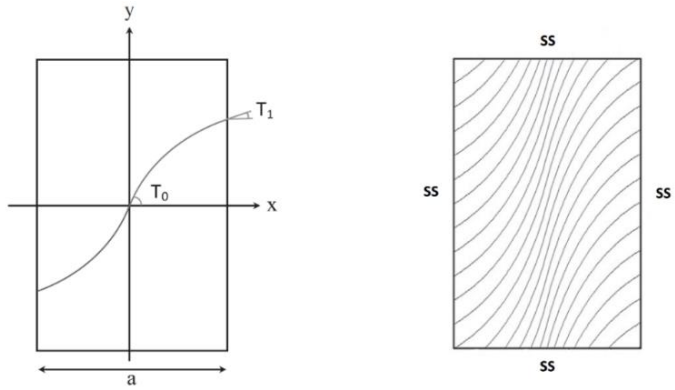

Figure 29: Curvilinear fibre path that varies linearly along the x-axis (left), and representation of the constraints with the fibres behaviour (right)

The mathematical description of the fibre variation behaviour in a single ply is reported in equation below [2]:

$$\vartheta(x) = \frac{2(T_1 - T_0)}{a}|x| + T_0 \tag{7.1}$$

where ϑ represents the fibre orientation, a denotes the plate width, T_0 and T_1 are the fibres angles at the plate centre ($x = 0$) and the plate edges ($x = \pm a/2$).

The plate is considered symmetric and balanced, composed by 16 plies. The symmetry of layers stacking sequence implies that the first 8 plies orientation angles patterns are repeated in an inverse order in the other 8, nullifying all the terms that couple the in-plane loads with the out-of-plane bending moments. A laminate is considered balanced if it has a layer with a negative ϑ orientation for every layer with a positive ϑ. This last assumption considered introduces a null shear-extension coupling terms.

The plate has the longer edge aligned along the x-axis. The dimension of the plate are 1200 mm and 1000 mm with a thickness of 0.125 mm per ply, made by IM7/8552 graphite-epoxy [55]. The material selected is widely implemented in aeronautical structures and all the engineering properties are reported in Table 10 and Table 11.

	Values
E_{11}	150000 MPa
E_{22}	9080 MPa
G_{12}	5290 MPa
ν_{12}	0.32
α_1	-5.5E-6 1/C°
α_2	25.8E-6 1/C°
ρ	1.55E-9 t/mm^3

Table 10: Composite material engineering properties

	Single ply	Double or outer ply
X_t	2323	2323
X_c	1200	1200
Y_t	160.2	101.4
Y_c	199.8	199.8
S	130.2	107.0

Table 11: Composite material ply strength properties expressed in MPa

The example optimization problems are solved with the implementation in MATLAB® of the various parts, i.e. the sampling of the lamination parameters, the evaluation of the input-output pairs with a Ritz's analysis, the built and the training process of the ANN, the optimization with the PSO algorithm and the post-processing conversion.

It has to be highlighted that all the operations in the work are done with an Intel® Core™ i7-6500U CPU @ 2.50 GHz with RAM of 12 GB DDR3 at 1600 MHz.

7.2 Bounds and sampling in the lamination parameters space

In this work, the lamination parameters are used directly as design variables. This is possible because the parametrization condenses the orientation angles information inside the lamination parameters sets. As already mentioned, a single set of LPs is not enough to characterize a VSP structure.

For the plate considered, the minimum number of sampling points is 2, generally taken at in the center and on one of the edges [2]. The linear fibres distribution, in fact, can be fully described by these two points, as reported in equation (7.1). As a reference, in the figure below where the points are sampled on the plate is shown:

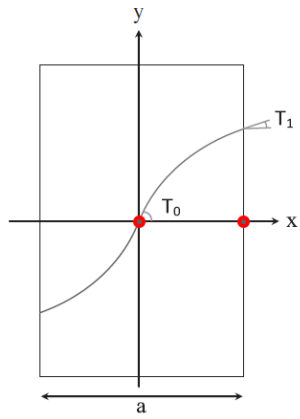

Figure 30: Sampling point on the laminate plate (red dots)

The minimum number of lamination parameters, considering the two-sampling point strategy for a VSP, is equal to 24. However, assuming a symmetric and a balanced plate the number of parameters is reduced. For symmetric plates

characterized by null membrane anisotropy, the lamination parameters simplify as $\xi_i^B = 0$ $(i = 1, \dots, 4)$ and $\xi_2^A = \xi_4^A = 0$ [16][22]. Therefore, due to the symmetry, the number of lamination parameters is reduced from 24 to 16, while due to the balance, other 4 lamination parameters should be neglected, obtaining a final number of 12 lamination parameters. Considering all the aforementioned assumptions, the lamination parameters set is composed by:

$$LP = \left[\xi_{1\,1}^A \quad \xi_{3\,1}^A \quad \xi_{1\,1}^D \quad \xi_{2\,1}^D \quad \xi_{3\,1}^D \quad \xi_{4\,1}^D \quad \xi_{1\,2}^A \quad \xi_{3\,2}^A \quad \xi_{1\,2}^D \quad \xi_{2\,2}^D \quad \xi_{3\,2}^D \quad \xi_{4\,2}^D\right] \tag{7.2}$$

where for each $\xi_{i\,j}^k$ the subscript j is referred to the point of the laminate, where 1 is for the center point and 2 for the edge point. The values of the lamination parameters in between the two sampling points are computed by interpolation. In this work the interpolation is done inside the Ritz's analysis code by a series of Legendre's polynomials.

The sampling of the lamination parameters is executed to generate sets that can be used for the training of the ANN. Therefore, a uniform distribution inside the design space is required to generate an appropriate training set for the ANN. The sampled sets of lamination parameters are passed through the Ritz's analysis code in order to obtain the input-output pairs needed.

The sampling procedure follows the method reported in Chapter 4, starting with an LHS, filtering, and saving only the sets that verify the constraints. The distributions of lamination parameters for three different starting number of points inside the design space are shown from Figure 31 to Figure 33. In all these figures, the lamination parameters are expressed with the notation V_i^k where the subscript i indicates the component, while the superscript k is related with the lamination constitutive matrices (**A**, **B** or **D**).

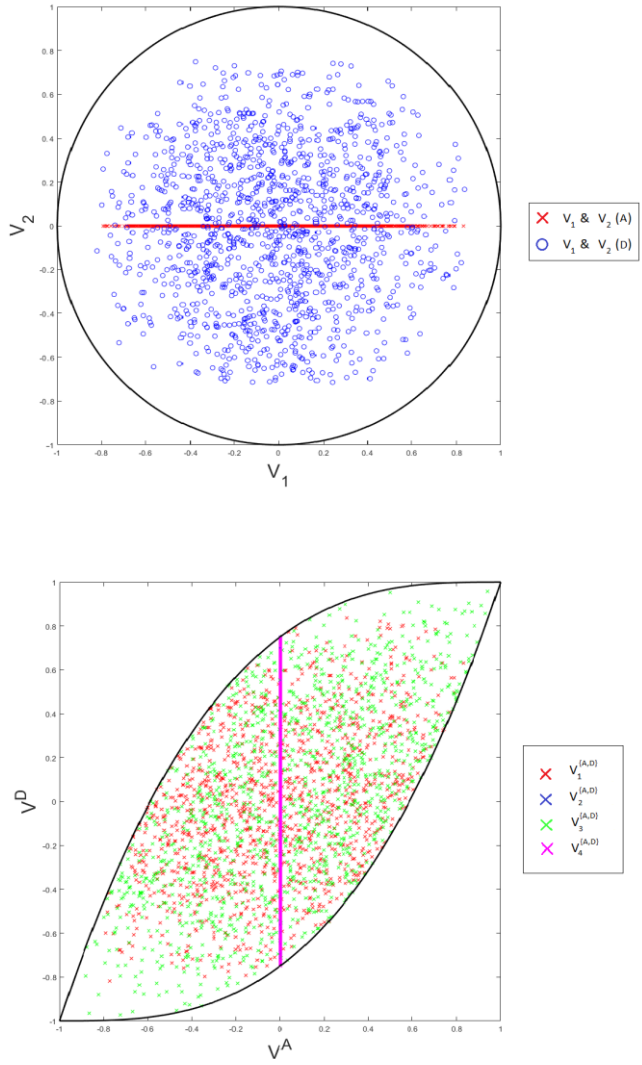

Figure 31: Distribution of first and second lamination parameters (up); distribution of lamination parameters accordingly with the constraints in equation (3.15)(bottom) (CASE 1)

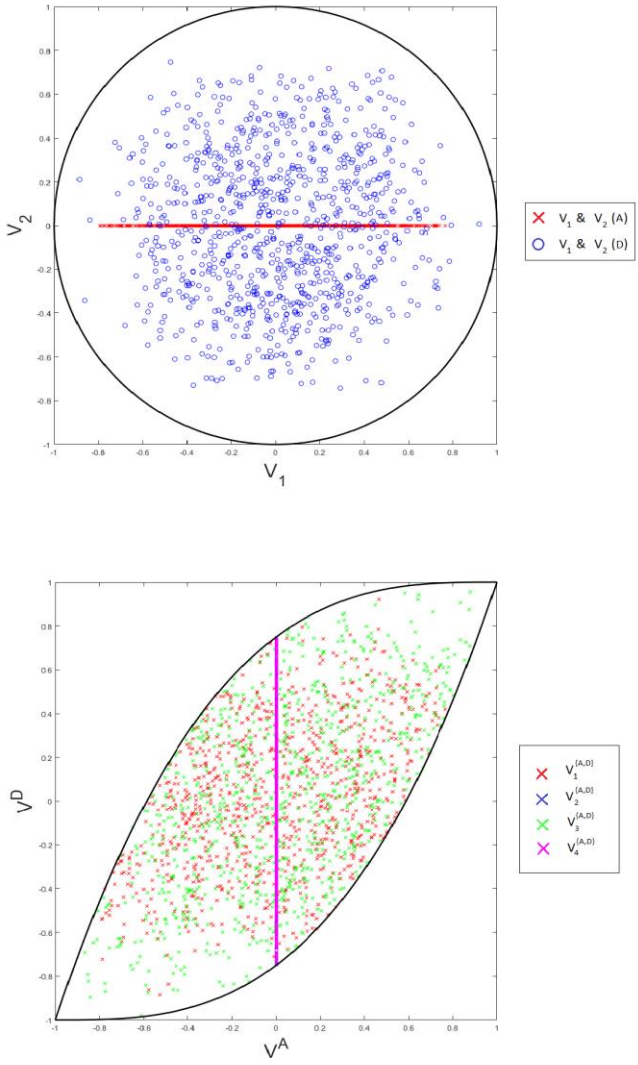

Figure 32: Distribution of first and second lamination parameters (up); distribution of lamination parameter accordingly with the constraints in equation (3.15)(bottom) (CASE 2)

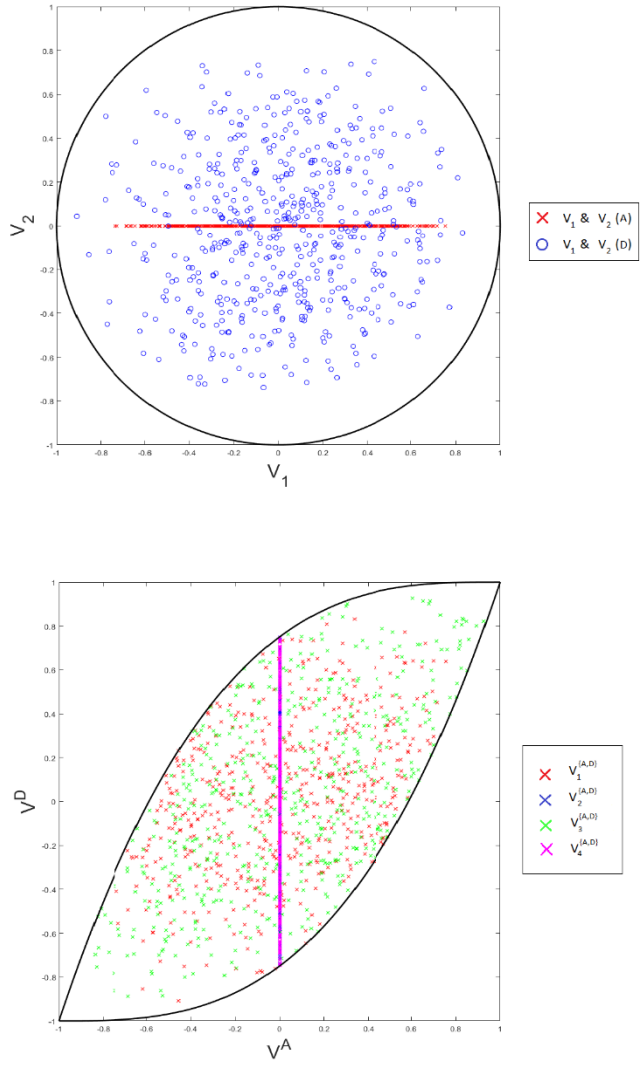

Figure 33: Distribution of first and second lamination parameters (up); distribution of lamination parameter accordingly with the constraints in equation (3.15)(bottom) (CASE 3)

The number of starting points and the final number of points are reported in Table 12. The starting point number refers to the initial number of sampled points in the LHS, while final point number refers to the final number of sets that verify the constraints in both points of the laminate.

	Starting point #	Final point #	Sample time [min]
Case 1	500000	1351	210.07
Case 2	350000	937	114.11
Case 3	200000	555	35.94

Table 12: Initial number of sampling points Vs final number of sampling points and time spent for the sampling procedure

The time spent to perform the sampling procedure increases more than linearly as the number of starting points increase. This happens because the twelve-dimensional space have to be subdivided in smaller intervals, increasing the possible combination and the number of points that have to be sampled.

The "Case 1" corresponds to the points shown in Figure 31, while "Case 2" is associated with Figure 32, and Figure 33 is linked with "Case 3". From these figures, one can observe that the central region is more densely populated and become denser as the number of sampled points is increased. The lamination parameters referred to the in-plane lamination constitutive matrix are more evenly distributed in space with respect to the out-of-plane matrix. This behaviour is due to the simplification done. In fact, the ξ_2^A (V_2 in the figures) have to be zero to generate a balanced laminate, placing all these lamination parameters on a horizontal line. This observation could be done also referring to the data reported in the bottom plots of Figure 31, Figure 32, and Figure 33 where ξ_2^A and ξ_4^A (V_2 and V_4, respectively) generate a vertical line and not a cloud of values.

Looking at the graphical representation of the data in figures from Figure 31 to Figure 33, most of the design space is well mapped, while the space near the boundaries have only a low number of points but is not unsampled.

The same procedure use to obtain the three cases in Table 12 is also used to sample three reduced sets. These sample sets are studied to be smaller in number of points

with a reduced time consumption. This second sets group is generated to perform the transfer-learning strategy. Therefore, this new training strategy improves the performance of the training in terms of time consumptions only if a small training set is used. In fact, with a set of the dimension reported in Table 12, the time saved is only during the training process, minimizing the improvements. The sampling data of the reduced sets are reported in the table below:

	Starting point #	Final point #	Sample time [min]
Reduced set 1	100000	271	7.74
Reduced set 2	50000	131	2.00
Reduced set 3	25000	72	0.45

Table 13: Initial number of sampling points Vs final number of sampling points and time spent for the reduced sets

In addition to the training sets, the testing sets has to be determined. The testing set is obtained by generating a random distributed set of lamination parameters inside the design space, checking that each one verifies the constraints. The dimension of the sets is fixed to be equal of the 20% of the relative training sets [34]. The relative number of testing and training sets are reported in Table 14.

	Training point	Testing point
Case 1	1351	270
Case 2	937	187
Case 3	555	111
Reduced set 1	271	54
Reduced set 2	131	26
Reduced set 3	72	14

Table 14: Number of training sets and number of the relative testing sets

The testing sets are sampled in the order of few seconds, making those times negligible, compared with the other time consumptions involved.

As anticipated, the training and the testing sets are put inside the Ritz's code in order to obtain the input-output pairs required. The time spent to execute the Ritz's analysis for the six sets differs on the nature of the example problem considered, and on the number of sets to evaluate. The time spent to obtain the input-output sets are reported in Table 15. It has to highlighted that the time reported is the sum of the ones spent to the training and the testing sets.

	Free Vibration - CPU time [min]	Buckling – CPU time [min]
Case 1	187.78	36.13
Case 2	136.94	27.56
Case 3	75.69	15.91
Reduced set 1	36.87	8.93
Reduced set 2	18.56	4.75
Reduced set 3	10.43	2.26

Table 15: CPU time for the generating input-output sets for both the training and the testing

From the data reported in Table 15, it can be observed that the reduced sets is not only less time consuming for what concerns the sampling process, but also for the numerical analysis via Ritz's method.

With the input-output pairs, the ANN can be trained and tested.

7.3 Design of the ANN

In this section are reported the procedures used in this work to obtain the input-output training and testing sets, and the tuning procedures to obtain the most efficient ANN. In fact, different aspects influence the efficiency of the neural network, needing a study of the number and configurations of the various parts.

In this work, two different artificial neural networks have to be designed to solve the example problems considered. MATLAB® is used both to build and train the artificial neural networks, allowing to build different architectures of ANN, choosing activation function, learning algorithm, and so on. The ANN features have been extended in the context of this thesis by developing new functions and capabilities. Specifically, the following features have been implemented: the testing process, the input file needed to start the built of the ANN and their training, and the heuristic training method. The design decisions are made both by studies done in this work and by relying on the literature.

The activation function is fixed, and the one selected is the Tanh. In fact, it is demonstrated that this function is one of the best solutions for and activation function that involves an optimization process with the use of an artificial neural network [38]. However, the output layer neuron is built with a linear activation function. Furthermore, on the base on the nature of the input points, the activation function selected is the identity one.

Another assumption that is done regards the number of hidden layers in the neural network. Most of the problem can be solved with an ANN built with only two hidden layers, varying the number of neurons inside them [6][7][33]. Finally, the biases are assumed to be present in each layer in order to simplify the network. In fact, MATLAB® allows to select in which layers introduce the bias and where do not.

The scheme of the ANN considered is reported in the figure below:

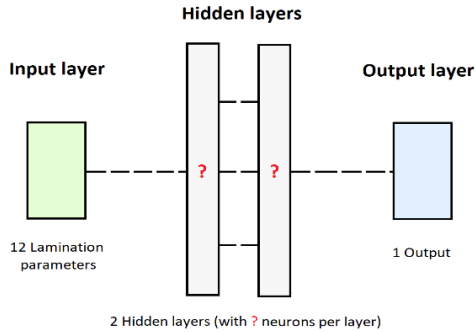

Figure 34: Simplified scheme of the ANN considered

In Figure 34 the dashed lines represent the interconnection between the different neurons in layers. These interconnections define the architecture of the ANN and determined yet. In order to simplify the work, the number of neurons inside hidden layers is considered equal. A study will be conducted to determine which is the number that maximizes the efficiency of the neural network. A neural network is more efficient than another if the percentual testing error is smaller and the training procedure consumes a comparable amount of time.

The number of neurons inside hidden layers is chosen by an iteration process. To avoid that the network become too heavy, the number of neurons considered ranges from 1 to 20. Therefore, the number of neurons inside the network, without considering the one inside the output layer, ranges from 2 to 40. Furthermore, two different kinds of topologies are considered to better analyse the performance of the network. The interconnection schemes chosen to be analysed are the feedforward topology and the cascade one. With all the aforementioned assumptions a set of different networks is build and all the ANN inside are trained in order to estimate which one gives the minimum percentual error on the results.

Another aspect to be established is the learning algorithm considered. In this work the LMb strategy is chosen because the dimension falls in size inside the feasible training domain for this kind of training algorithm. Furthermore, this strategy avoids possible lack of convergence due to the complexity of the example problems considered [44]. As reported in section 4.2.3, the second order algorithms are dependent on the starting points of the parameters. MATLAB® in the "*train"* function, used to the training procedure of the network, select randomly the starting initial values unless their values are introduced manually. To avoid complex analysis that determine the most efficient starting point, a heuristic strategy is considered.

The process is composed by different training phases built in series. In each iteration step, the training is executed in standard way using the LMb algorithm, and the relative testing and training percentual errors are evaluated. At the end of the numerical iteration, the configuration of weights that minimises the testing percentage errors (average and maximum) is saved. Thus, the heuristic strategy does not select the most efficient starting points but selects each time different initial positions of the weights and evaluate the errors associated with. Therefore, a large number of starting values are considered avoiding lack of convergence. The procedure is outlined in Figure 35.

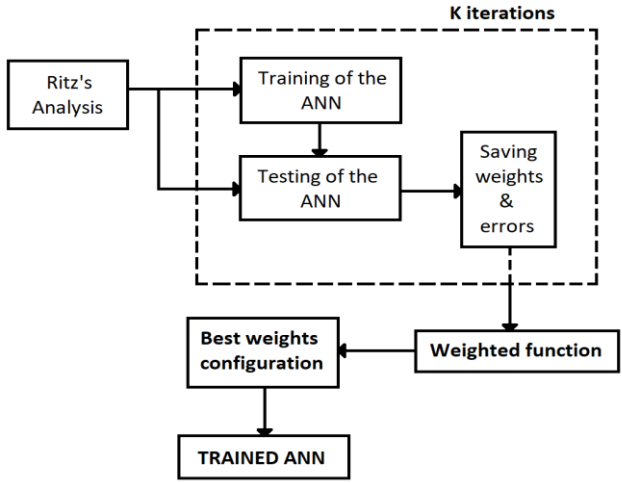

Figure 35: Procedure scheme for the ANN training

This procedure allows to obtain the same order of error every time, making the training procedure accurate and reliable. However, the number of iterations has to be determined in order to achieve convergence in a reasonable amount of time. Discrete values for the number of iterations are considered, i.e. 5, 10, and 20, where for each ones the code is executed five times in order to investigate on the convergence. From the data obtained by the iteration process, 20 iterations grant a level of convergence that is acceptable. In fact, every time that the training of the ANN is processed, the final testing percentage errors are always in the same ranges of values, with a small gap between the end values.

All the analysis for the number of neurons and architecture are executed, considering the three sampled sets of input-output pairs reported in Table 12.

7.3.1 Training error

The maximum and the average training percentage error for the free vibration problem are reported in the figures below, distinguishing the one obtained with a feedforward network and the ones obtained with the cascade one:

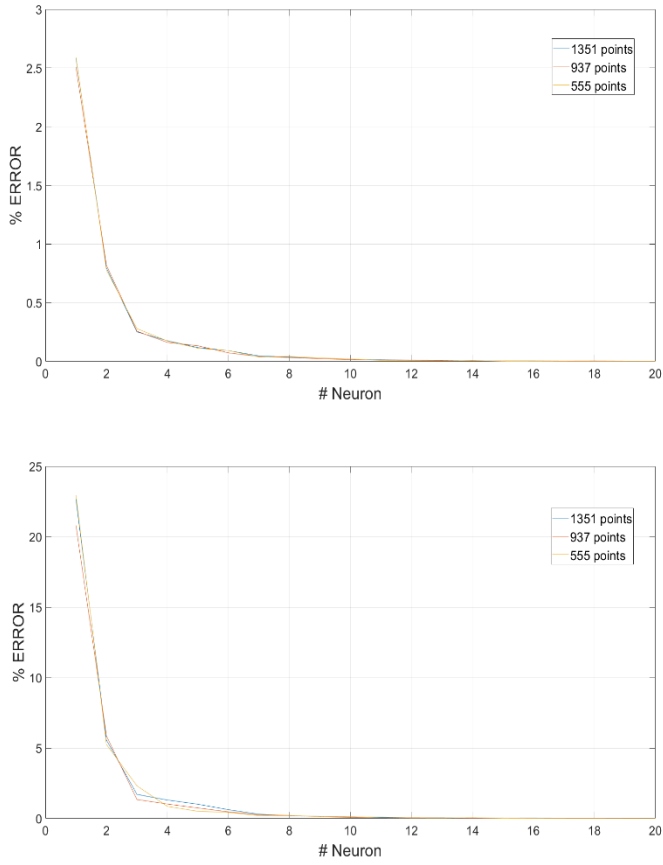

Figure 36: Average percentage error in the training process (up), and Maximum percentage error in the training process (bottom) for the free vibration problem with a feedforward network

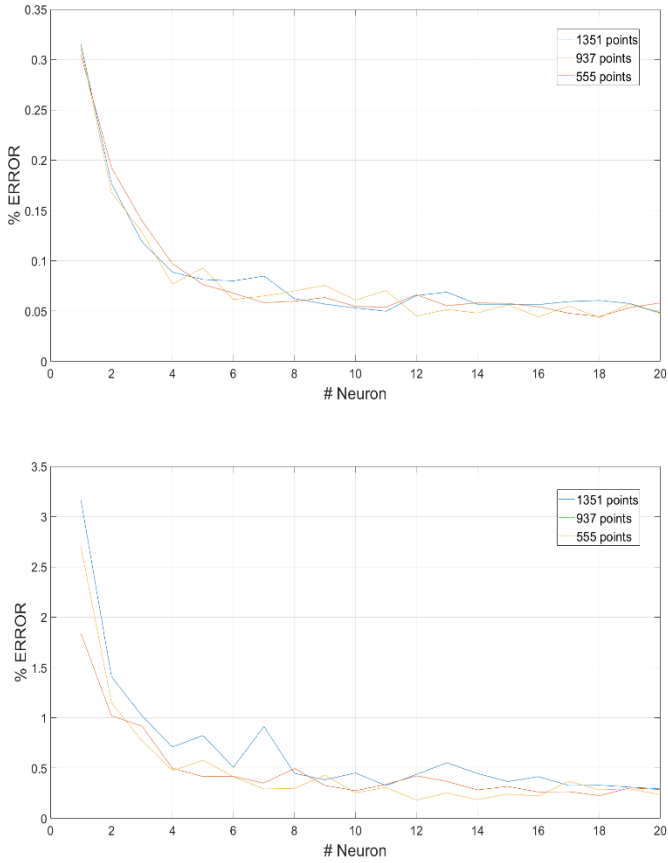

Figure 37: Average percentage error in the training process (up), and Maximum percentage error in the training process (bottom) for the free vibration problem with a cascade network

From Figure 36 and Figure 37, it can be observed that for both the topologies, as the number of neurons increases, the percentual error decreases.

For what concerns the buckling problem, the maximum and the average training percentage error are reported in Figure 38 and Figure 39.

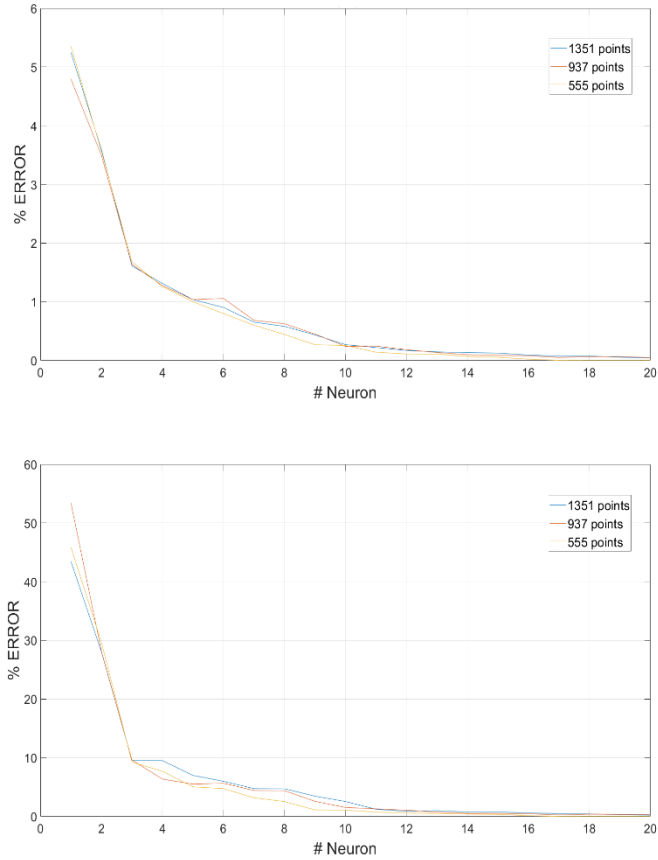

Figure 38: Average percentage error in the training process (up), and Maximum percentage error in the training process (bottom) for the buckling problem with a feedforward network

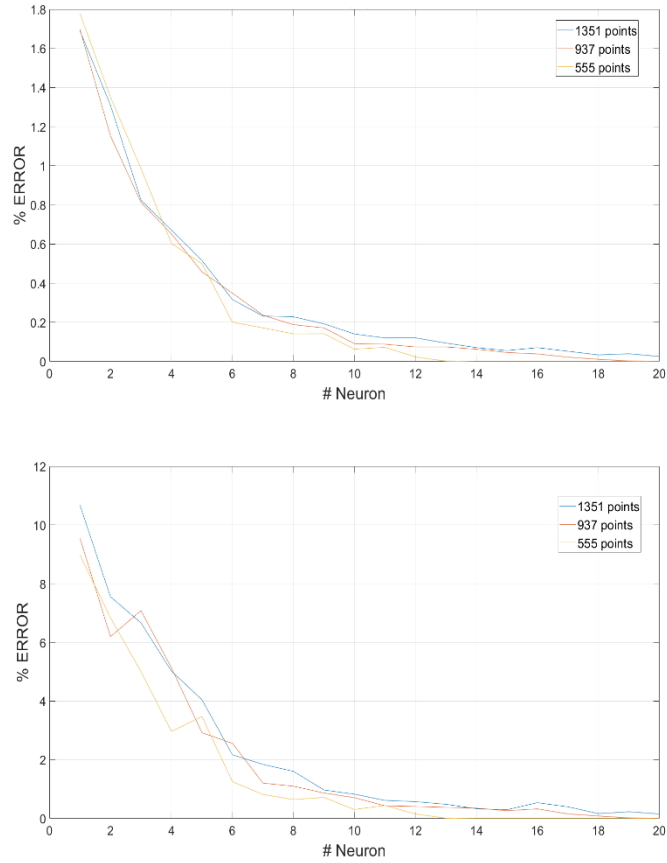

Figure 39: Average percentage error in the training process (up), and Maximum percentage error in the training process (bottom) for the buckling problem with a cascade network

Also for the buckling problem the behaviour of the training error follow the same trend. Therefore, the percentual error tends to zero as the number of neurons per layer increases. However, the over-fitting phenomenon has to be avoided, needing an investigation also for the percentage error on the testing sets.

7.3.2 Testing error

The average and the maximum percentage testing errors are evaluated. These errors are used to select the best architecture between the selected ones. The percentage testing error for the free vibration problem in function of the number of neurons per layer is shown in the figures below, reporting both the ones obtained with a feedforward architecture and a cascade one:

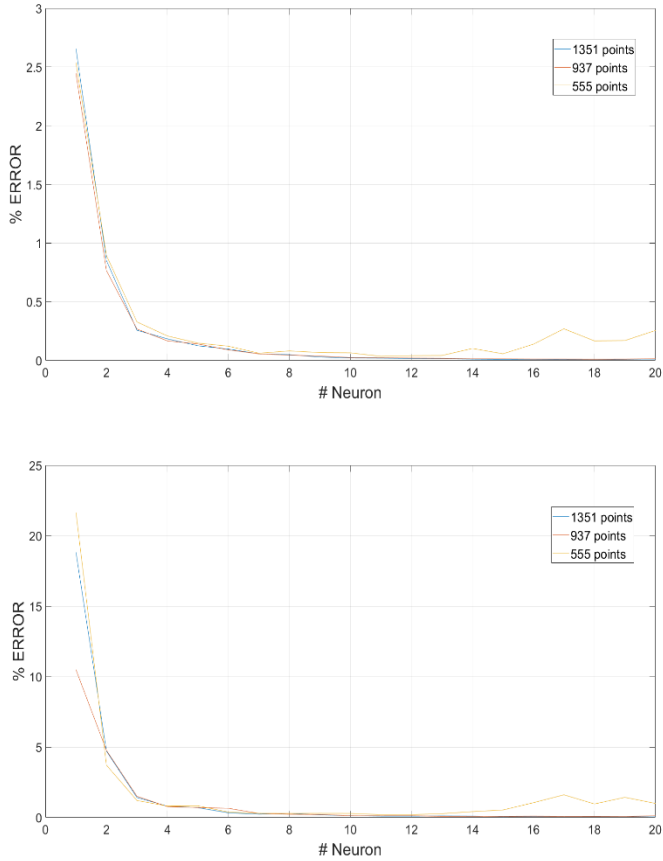

Figure 40: Average testing percentage error (up), and Maximum testing percentage error (bottom) for the free vibration problem with a feedforward network

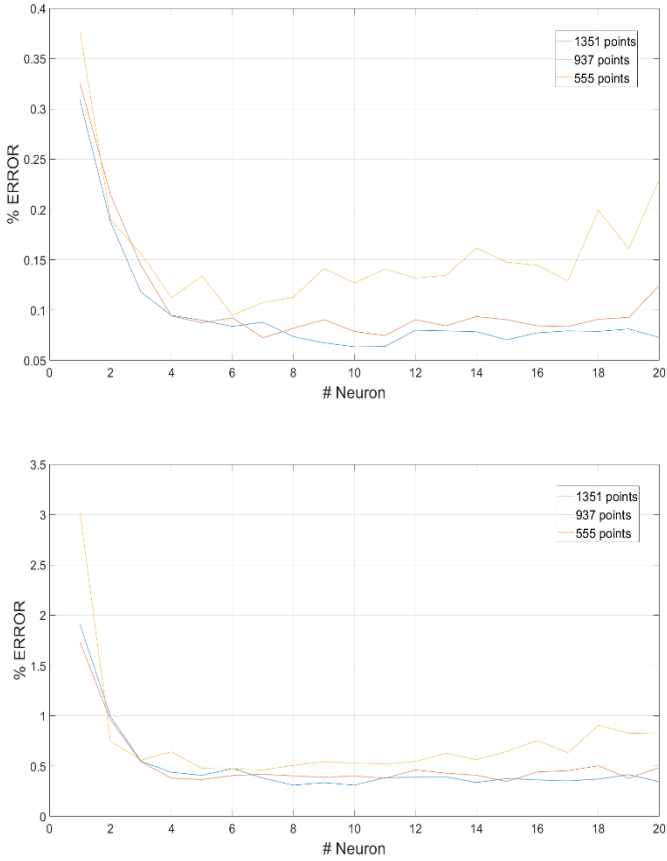

Figure 41: Average testing percentage error (up), and Maximum testing percentage error (bottom) for the free vibration problem with a cascade network

The behaviour shown in Figure 40 and Figure 41 is comparable with the one shown in the reference Figure 23. In fact, from 11 neurons the behaviour shown an over-fitting problem, with the loss of generality. Same trend can be observed in the

following figures reporting the average and the maximum testing percentage error for the ANN associated with the buckling problem:

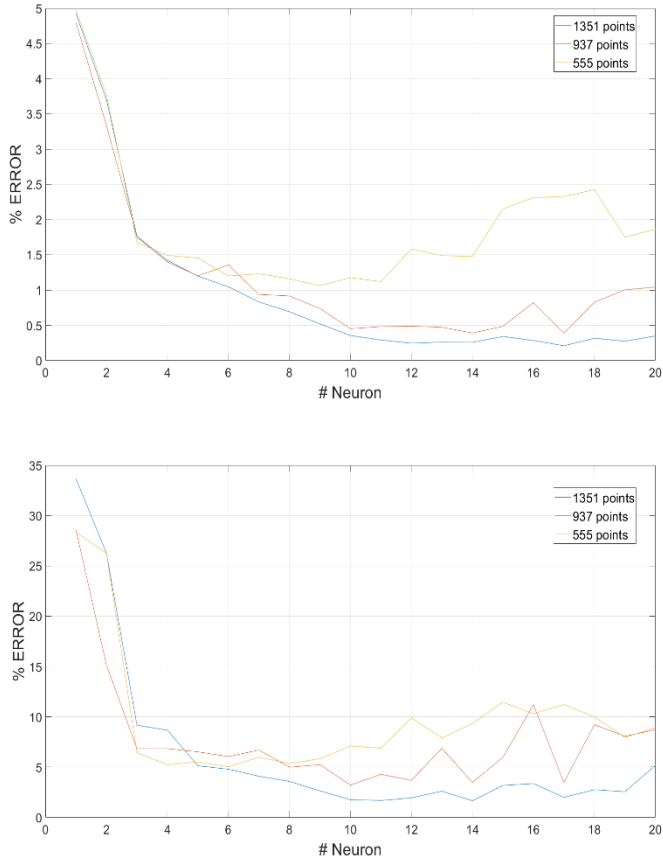

Figure 42: Average testing percentage error (up), and Maximum testing percentage error (bottom) for the buckling problem with a feedforward network

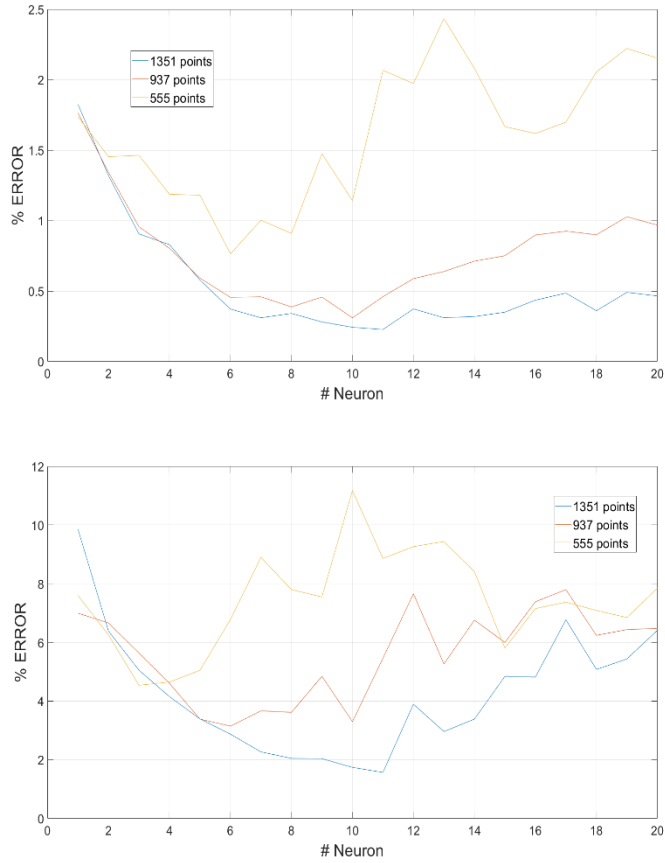

Figure 43: Average testing percentage error (up), and Maximum testing percentage error (bottom) for the buckling problem with a cascade network

In these cases, an increase of error can be noticed for a number of neurons higher than 8. It is believed that this behaviour is due to an overfitting issue.

The time spent on the training phase is also a focal point that has to be taken into consideration and moves the decision on one ANN configuration rather than another. The different time consumption depending on the number of neurons, the

example problem of reference, and the architecture of the network are reported in the following figures:

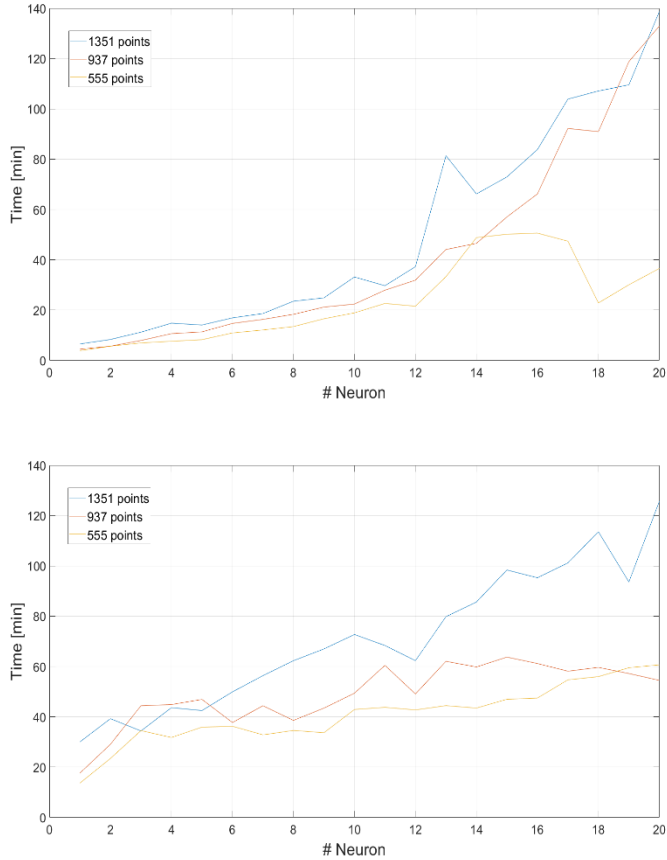

Figure 44: Time consumption with a feedforward network (up), and with a cascade network (bottom) for the free vibration problem

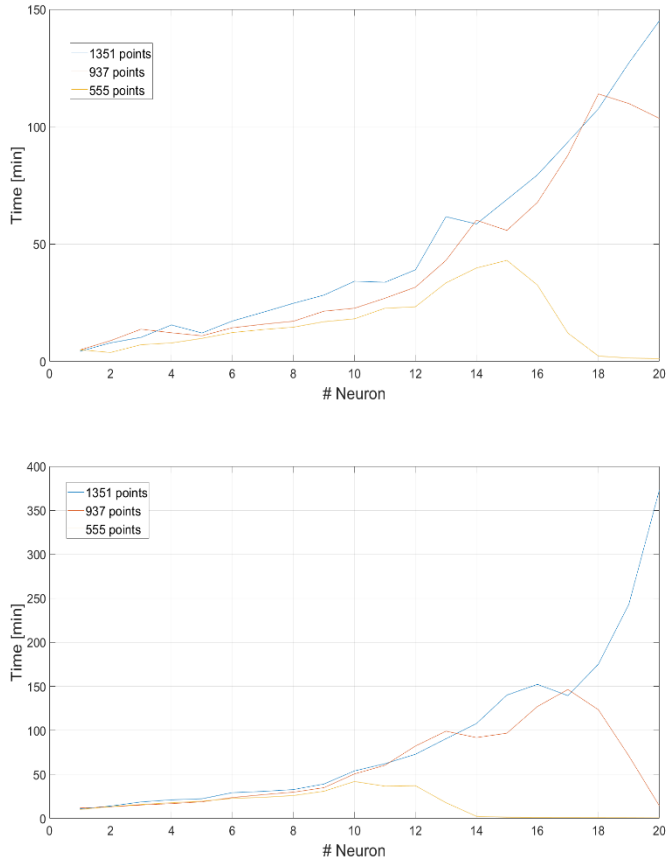

Figure 45: Time consumption with a feedforward network (up), and with a cascade network (bottom) for the buckling problem

7.3.3 Selection of the ANN

The foremost choice to make is related to the network topology. For what concerns the free vibration problem, the network with the feedforward strategy is clearly more performing both in terms of time spent for the training phase, and the testing

percentage errors. Therefore, the feedforward topology is selected for the free vibration example problem.

On the other hand, the accuracy in predicting the buckling load is similar for the two architectures, with relevant differences regarding the training time. The testing percentage errors related to the average for the cascade network is slightly better than the one referred to the feedforward one. On the contrary, the maximum testing error is lower for a feedforward topology. Considering that the difference in time is relevant only for a large number of neurons per layer, the factors that determine the selection are restricted only on the testing percentual errors. The cascade network has smaller error until the 11 neurons, presenting over-fitting problem with a larger number of neurons. The feedforward network, instead, presents larger errors until the 11 neurons, obtaining the appearance of the over-fitting phenomenon only from 14 neurons.

Starting from the assumptions that the network has to be built with the smaller number of neurons possible, the cascade network topology is chosen for the buckling problem. Therefore, it grants an acceptable level of accuracy with a small number of neurons.

In order to evaluate the performance of the ANN, an upper limit has to be imposed both on time and on the percentage errors. The maximum error allowed is about 5%, while the consumption of time is preferable that is under 50 minutes. From these considerations the number of neurons per layer is determined. Therefore, a weighted function is introduced where the two types of errors and the time spent for the training are summed up. The weights in this function tend to increase in dependence of the value of the errors and the time spent. The weight related to the errors is unitary until the value of the error overcome the 5%, becoming five times itself; the one referred to the time consumption is fixed to unitary value if the time is more than 50 minutes, otherwise it is thousand times smaller. The code of the functions that determine the value of the weights is reported below:

```
If MaxTestErr ≥ 5
    w_a = 5;
else
    w_a = 1;
end
```

```
If time > 50
    w_t = 1;
else
    w_t = 0.0001;
end
```

The most promising number of neurons per layer for the three different sets considered for the free vibration problem is reported in the following table:

	# Neurons	AVERAGE % error	MAX % error	Time [min]
Case 1	11	0.2908	1.6833	33.71
Case 2	14	0.0137	0.0721	46.55
Case 3	12	0.0439	0.2184	27.27

Table 16: Neural network that minimize the weighted function and the relative parameters (errors and time) for the free vibration problem

The highest errors are obtained for "Case 1", followed in order by "Case 2" and "Case 3". Neglecting "Case 1" because of the higher errors, the other two are compared. The most efficient network is the one built with the data referred on "Case 3". In fact, in "Case 3" the time spent is about 27 minutes, against 46 minutes, saving half of the time. For what concerns the percentage errors, "Case 3" and Case 2" are in the same order of magnitude. These differences do not justify the possible choice of the "Case 2" network, that wastes time to have an imperceptible improvement.

The results obtained for the buckling problem ANN that have to be considered are:

	# Neurons	AVERAGE % error	MAX % error	Time [min]
Case 1	9	0.2801	2.0374	39.11
Case 2	6	0.4851	3.3391	21.98
Case 3	3	1.4631	4.5376	15.89

Table 17: Neural network that minimize the weighted function and the relative parameters (errors and time) for the buckling problem

The errors for "Case 3" are close to the threshold of 5%, whilst the advantages of "Case 2" against "Case 1" are clear by comparison of the errors and the total time.

The topology, the number of neurons per layer, the training sets dimension, and the time spent that were selected, are reported in Table 18. The "Total time" refers to the sum of training, sampling, and numerical Ritz's evaluation times, reported in Table 16 and Table 17, in Table 12, and in Table 15, respectively.

	Topology	# Hidden layers	# Neuron per layer	Training set dimension	Total time [min]
Free Vib.	Feedforward	2	12	555 (Case 3)	138.90
Buckling	Cascade	2	6	937 (Case 2)	163.65

Table 18: Neural network characteristics for the free vibration problem and for the buckling problem

The generalization characteristics that the ANN shows with the small percentage testing error, grants that analytical analysis or numerical ones can be replaced with ANN at the cost of a small error increment.

7.3.4 Transfer-learning results

The strategy adopted consists in a transfer-learning from the ANN associated to the free vibration problem to the one associated with the buckling problem and vice versa. This implies that the network that is the most efficient in one case is used also for the other kind of problem.

The data obtained from a transfer-learning procedure are reported in Table 19. The expression "F → B" and "B → F" are referred to the two different starting point of the transfer learning process. The first one is associated with an ANN trained and built to solve the free vibration problem where its weights are taken and used as starting point for the training of the buckling problem ANN. The second one uses the weights of the ANN associated to the buckling problem as starting point for the training of the ANN built to solve the free vibration problem. The data needed to execute a comparison with the Most Efficient ANN (ME) are reported. These errors are associated with the second ANN, the one trained with the transfer-learning (second letter in the notation used).

		F → B	B → F
	(ME) MAX % err	3.3391	0.2184
	(ME) AVG % err	0.4851	0.0439
RS1	Max % error	13.4184	2.2523
	Average % error	3.0044	0.3582
RS2	Max % error	17.0612	2.7182
	Average % error	4.5657	0.6404
RS3	Max % error	23.0982	9.2736
	Average % error	5.7002	2.6962

Table 19: Maximum and Average testing error with different reduced sets and different strategy for the transfer-learning procedure

where "RS1", "RS2", and "RS3" are the number of reduced training sets as reported in Table 13.

From the data reported in Table 19 one can observe that the performances of the transfer-learned ANN are lower respect the one built with the characteristic expressed in Table 18. However, the percentual errors referred to an artificial neural network trained with the weights coming from an ANN built to solve the buckling problem (B → F), are acceptable. The ANN associated to the free vibration problem is the one with the smaller testing errors. For this reason, even if the training algorithm is the transfer-learning one, the level of error tends to remain below the one obtained by the "buckling ANN". Therefore, the ANN trained so far could be used in substitution of the one trained with standard strategies.

The time spent in the transfer-learning training process are reported in Table 20. The time difference is computed as expressed in the following equation:

$$Time\ difference = Training\ time\ ME - Training\ time\ RS \tag{7.3}$$

		F → B	B → F
Time difference [min]	RS1	21.49	12.92
	RS2	21.84	27.12
	RS3	21.87	27.18

Table 20: Difference in training time between the classical training procedure and the transfer-learning one

From the data expressed in Table 19 and Table 20, it can be observe the differences between the standard training procedure and the transfer-learning one. In fact, for the transfer-learning strategy, the percentual errors are larger, but the time spent for the training is lower.

Comparing the percentual errors and the time consumption the best solution between the proposed ones is the strategy that employs the "RS2" reduced sampling set.

The shortcoming of this strategy is an increase of the maximum and average testing percentage errors of about 2.5% and 0.31%, respectively. The advantage is a net time saving in the order of 61.06 minutes, considering not only the difference in time between the training phases, but also the spared sampling time, which is about 33.94 minutes, as reported in Table 12 and Table 13. The time consumption becomes more relevant as the complexity of the problem increases.

It shall be underlined that in the following parts where the free vibration problem is considered, the artificial neural network used is always referred to an ANN trained with transfer-learning.

7.4 PSO internal parameters tuning

The PSO algorithm is implemented in the process in order to find the optimal value for the two example problems. The ANN obtained in the previous section (7.3) are transformed into a MATLAB® function that works as objective function for the *particleswarm* algorithm. Furthermore, the penalty terms shown in equation (5.7) is implemented to find the optimal combination that maximizes the first natural frequency and the first buckling load. The penalty value M are assumed to be in the

form reported in equation (5.8), with the scalar a that assumes a value of 100 in order to bring the error e to a magnitude that can significantly modify the value of the objective function $f(\boldsymbol{x})$.

The error e is the maximum error obtained checking the lamination parameters constraints reported in Chapter 3. In order to verify properly the constraints, the code that fully rebuilt the lamination parameters set is implemented in the objective function, introducing the terms omitted by the simplifications done in section 7.1.

The trust parameters, the number of particles in the swarm, and the maximum number of iterations (stop criterion) are studied.

7.4.1 Trust parameters

A particular iteration is done to identify the set of trust parameters that gives as an output the best results. In this part of the study the opposite dual problem is used. The best configuration of trust parameters for the two different example problems are chosen as the ones that grant the smallest negative values.

It has to be highlighted that in this study, the maximum number of iterations is fixed to 3000 and the initial swarm is composed by two times the dimension of the design variables set, that in this particular case is 12.

During the optimization of the trust parameters, a square discrete domain $[1.49\ 2.05]^2$ is considered, containing 100 elements in it. The optimal values obtained with different configurations of trust parameters for the first natural frequency and the first buckling load are reported in Figure 46 and Figure 47, respectively.

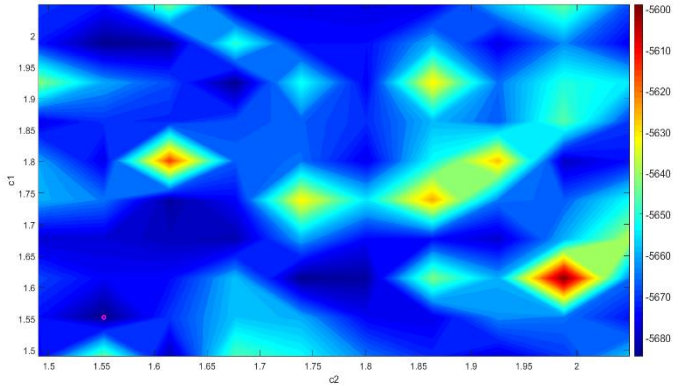

Figure 46: Variation of the value of the objective function as the trust parameters change for the free vibration problem

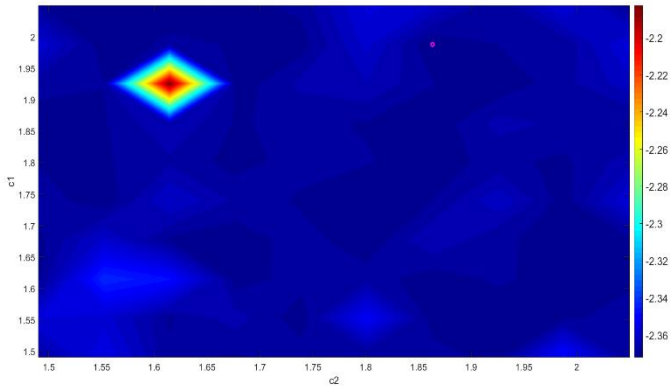

Figure 47: Variation of the value of the objective function as the trust parameters change for the buckling problem

The combination that produces the best results are highlighted in Figure 46 and Figure 47, where the corresponding values are reported in Table 21.

	Free Vibration	Buckling
C_1	1.5522	1.9878
C_2	1.5522	1.8633

Table 21: Trust parameters reported in equation (4.2) for the free vibration problem and for the buckling problem

Fixing the values of the trust parameters implies that each particle does not change the behaviour of its movements in the feasible design space. Therefore, each particle weights the knowledge of the swarm and itself confidence in the same way during the entire optimization process.

7.4.2 Dimension of the swarm

The initial swarm is chosen as done for the testing set for the artificial neural network, as reported in Chapter 4. Therefore, the starting design variables values of the particles are selected randomly inside the feasible design space.

For this part of the study, the number of iterations remains 3000, while the trust coefficients are the ones obtained in section 7.4.1. The optimal number for the swarm dimension is obtained implementing a weighted function, where the time spent to the optimization searching procedure is added to the standard deviation. This is done to consider the time spent for the optimization process that increases as the number of particles increases. The standard deviation is chosen as a core parameter to try to identify the smallest swarm that grant the desired level of convergence. The standard deviation is evaluated on a set of 10 optimal results obtained fixing all the parameters, including the initial swarm, evaluating in this way as the results oscillate.

The swarm is created with the biggest number of particles (10n). This allows that at each iteration of the study process, the swarm is not generated from zero. Therefore, the number of particles in the swarm increases respect to the previous iteration block, adding new points to verify better the performance of the dimension of the swarm and not of the swarm itself. With this strategy the swarm is every time the same with only the adding of new particles.

Two different iteration processes are performed to determine the most efficient number of particles that build up the swarm for the free vibration and buckling problems. The results of the two iteration processes are shown in the following figure:

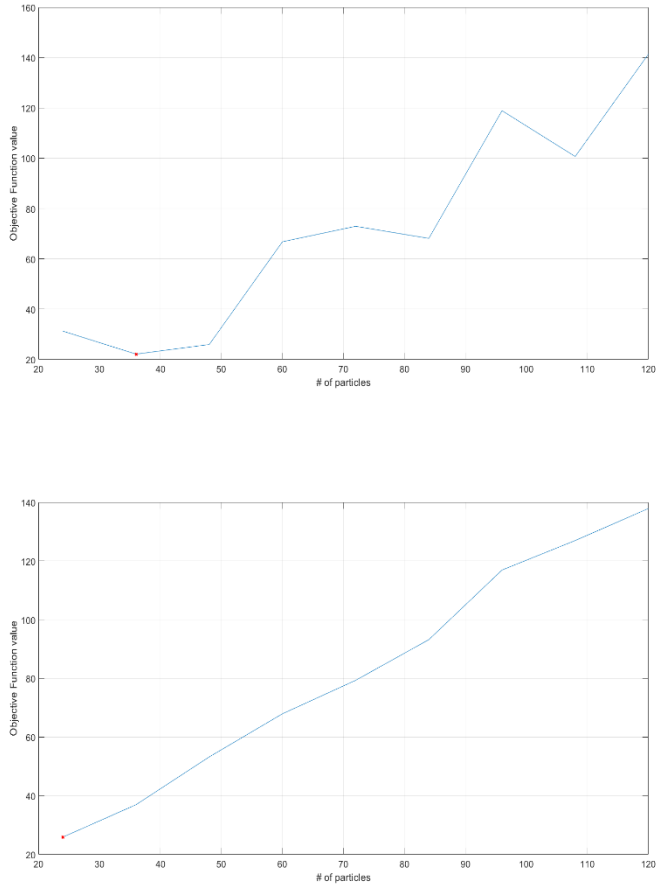

Figure 48: Variation of the value of the modified objective function as the number of the particles in the swarm increases for the two problem (free vibration up, buckling bottom)

In Figure 48, the more efficient number of particles inside the swarm is highlighted. It can be observed that for the buckling problem the optimal dimension of the swarm correspond to the minimum one (2n = 24), while for the free vibration problem the optimal value is three times the number of input variables (3n = 36).

7.4.3 Maximum number of iterations

In MATLAB® the *particleswarm* function starts with a default value of 1000 maximum iterations but it will be demonstrated not sufficient to reach the minimum value in the free vibration problem as in the buckling problem. Therefore, the analysis of section 7.4.1-2 have 3000 maximum iterations.

The number of iterations is augmented to 5000 and at this point, although the convergence is not yet ensured, the trend seems stable. For what concerns the trust parameters and the dimension of the swarm, the best solution obtained in the studies of this section are implemented in this analysis.

The behaviour of the optimal values through the iteration process of the PSO algorithm are reported in Figure 49 and in Figure 50, for the free vibration problem and for the buckling one, respectively.

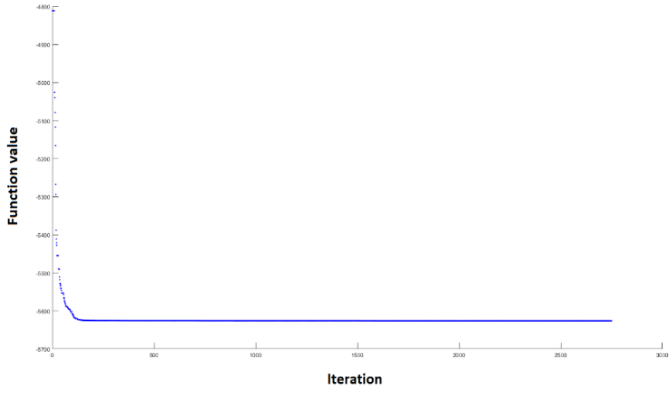

Figure 49: Variation of the optimization function value in function of the number of the number of iterations in the free vibration problem

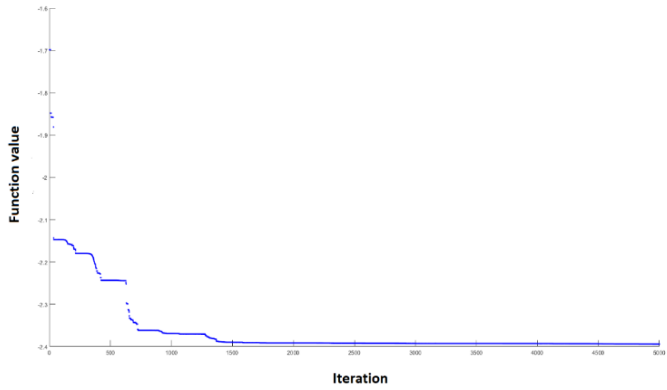

Figure 50: Variation of the optimization function value in function of the number of the iterations in the buckling problem

In Figure 50 is shown that the behaviour of the optimum value, after 3000 iterations, displays negligible improvements. Lowering the value from 5000 to 3000 could help sparing some computational time, that for 5000 iteration is about 190 seconds. Different consideration could be done for the free vibration problem, as shown in Figure 49. In fact, after 2750 iteration the value of the optimal solution tends to remain stable stopping the algorithm, hypothesizing that the convergence is reached. In this last problem the time spent for the optimization procedure is equal to 78.08 seconds.

All considered, the maximum number of iterations is fixed to 5000 as the algorithm performs rather fast compared to the other blocks of the procedure. For PSO computation the time spent is in the order of a few minutes, while for the determination of the inputs set and for the training of the artificial neural network the time consumption is in the order of several minutes or even hours.

7.4.4 Selection of the dual problem strategy

In this subsection, a comparison analysis is done on the output of the neural network. In fact, the dual problem has to be generated in order to find the maximum. The aim of this procedure is to evaluate if there exist significantly

changes in the performance of the PSO if the inverse is implemented in the code instead of the opposite.

Ten iteration are done for each of the two example problems, where each iteration compares the optimal results found by the PSO using an opposite objective function value or an inverse objective function value. All the internal parameters of the PSO are the ones obtained in this section.

The maximum difference using one strategy instead of the other one for a free vibration problem is in the order of 0.5%, while the average difference is under the 0.02%. For what concerns the buckling problem, the maximum and the average difference in values are 0.2% and under the 0.03%, respectively. It can be observed that the use of the opposite or the inverse is irrelevant in terms of efficiency, using one or the other without noteworthy losses. However, the opposite strategy is implemented in the code to avoid possible numerical errors making the inverse of small values, as for the first buckling load.

7.5 PSO results

In this section the results obtained from the PSO are reported. Furthermore, the optimal configuration lamination parameters obtained with the PSO are introduced inside a code that performs a Ritz's analysis. The configuration sets of the two example problems obtained from the PSO are reported in Table 22.

	Free Vibration		Buckling	
	centre	edge	centre	edge
ξ_1^A	-0.5969	-0.4912	-0.0284	-0.2574
ξ_2^A	0	0	0	0
ξ_3^A	-0.2874	-0.5174	-0.2131	-0.8675
ξ_4^A	0	0	0	0
ξ_1^B	0	0	0	0
ξ_2^B	0	0	0	0
ξ_3^B	0	0	0	0
ξ_4^B	0	0	0	0
ξ_1^D	-0.4723	-0.0858	-0.1901	0.0846
ξ_2^D	-0.1512	0.1751	0.0328	-0.0438
ξ_3^D	-0.5481	-0.9719	-0.8781	-0.9856
ξ_4^D	0.2359	-0.1898	0.2874	-0.0194

Table 22: Lamination parameters sets coming from the PSO optimization process

The values obtained from the optimization process and the ones obtained from a numerical analysis, with the percentual difference between the two, are reported in Table 23.

	PSO	Ritz	% error
Free Vibration	11.94 Hz	11.87 Hz	0.5545
Buckling	2.39 N/mm	2.37 N/mm	0.8368

Table 23: Comparison between the PSO results and the results obtained from a Ritz's analysis, reporting the percentual errors between the two

From the data reported in Table 23, it can be observed that the PSO procedure gives results very similar to the ones obtained from a Ritz analysis. The errors are introduced by the approximation executed by the ANN. The percentual error in the free vibration problem (first natural frequency), as for the one of the buckling problem (first buckling load) is less than 1%. The small difference between the values obtained in the two type of analysis make the whole optimization process reliable.

7.6 Post-processing internal parameters and results

The objective function is built in order to ensure the symmetry of the laminate, obtaining null errors on the lamination parameters referred to the coupling laminate constitutive matrix (**B**).

The weights of the objective function are obtained using the Ritz's code, evaluating the strain energy. The value of the energy terms and their relative weights for the free vibration problem and the buckling one, are reported in Table 24.

	Free Vibration	Buckling
U_A	0 J	20982.51 J
U_B	0 J	0 J
U_D	3.1556 J	0 J
w_a	0.5	2.0983
w_b	0.5	0.5098
w_d	3.1556	0.5098

Table 24: Strain energy terms (Joule), and weights for post-processing analysis referred to the laminate constitutive matrices

The internal parameters of the PSO used for the post processing are the ones obtained in section 7.5. However, the most efficient swarm dimension is studied. The maximum dimension of the swarm is increased, because a swarm with 10 times the number if inputs does not grant a sufficient accuracy. Therefore, the largest number of particles in the swarm is fixed to be equal to 200n, while the minimum is set equal to 2n, where n is the number of design variables.

The study is executed considering a weighted function, where the standard deviation and the time spent are summed. The standard deviation considered is the maximum one between the standard deviation of the three different terms of the lamination parameters set. In fact, the three standard deviation are computed separately in order to split the three structural behaviour of the laminate.

The values of the weighted function that implement the standard deviation and the time consumption in function of the number of particles inside the swarm, are shown in Figure 51.

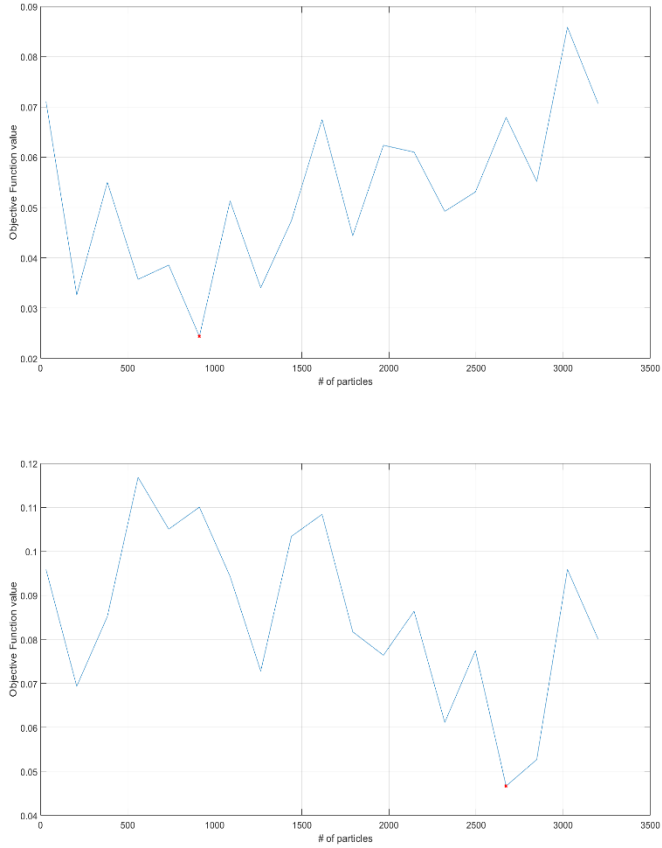

Figure 51: Variation of the value of the weighted function as the number of particles in the swarm increases for the two problems (free vibration up, and buckling bottom)

For the free vibration problem, the smaller number of particles in the swarm is computed to be 57 times the number of inputs. For what concerns the buckling problem, the swarm is composed by 167 times the number of inputs.

It can be observed that the dimension of the swarm referred to the buckling problem is very large, but the value of the objective function is clearly lower with a small difference in time consumption (about 1 minute). Also, for swarm referred to the free vibration problem the number of particles that is defined as the most efficient one is over the 10 times the number of design variables.

It has to be underlined that for example a value of 0.1 in the standard deviation for values that are in the range of [-1;1] is too large to be accepted.

7.6.1 Post-processing results

The lamination parameters expressed in Table 22 are the starting point for the transformation of the post-processing phase and are also the target lamination parameters present in equation (6.1). The angles obtained from the post-processing conversion are reported in Table 25.

PLY	FREQUENCY		BUCKLING	
	centre [°]	edge [°]	centre [°]	edge [°]
1st & 16th	53.63	48.48	45.24	41.29
2nd & 15th	-59.66	-42.65	-55.76	-43.78
3rd & 14th	-58.73	49.28	37.60	50.25
4th & 13th	-57.18	-45.56	-53	-48.04
5th & 12th	90	51.57	-28.68	55.06
6th & 11th	57.89	-55.88	2.73	-53.80
7th & 10th	57.44	83.01	68.31	57.37
8th & 9th	-89.99	-78.82	90	-58.36

Table 25: Orientation angles of the plies of the laminate evaluated in the center of the plate and on the edge for the free vibration problem and the buckling problem

The orientation angles obtained generate sets of laminate parameters that differ from the target ones. The errors are not percentual errors, because the lamination

parameters values are small, and a different kind of representation could generate meaningless or incomprehensible data. The errors of every terms of the lamination parameter sets for each kind of problem are reported in Table 26.

	FREQUENCY		BUCKLING	
	centre	edge	centre	edge
$\xi_1^A - \xi_{1_t}^A$	0.0325	0.1536	0.0451	0.0536
$\xi_2^A - \xi_{2_t}^A$	0.0115	0.0136	0.0019	0.0022
$\xi_3^A - \xi_{3_t}^A$	0.0566	0.0249	0.0155	0.0300
$\xi_4^A - \xi_{4_t}^A$	0.0389	0.0223	0.0035	3.97E-4
$\xi_1^B - \xi_{1_t}^B$	0	0	0	0
$\xi_2^B - \xi_{2_t}^B$	0	0	0	0
$\xi_3^B - \xi_{3_t}^B$	0	0	0	0
$\xi_4^B - \xi_{4_t}^B$	0	0	0	0
$\xi_1^D - \xi_{1_t}^D$	0.0257	0.0079	0.1729	0.1191
$\xi_2^D - \xi_{2_t}^D$	0.0297	4.65E-4	0.0759	0.2227
$\xi_3^D - \xi_{3_t}^D$	0.0086	0.0397	0.1202	0.0414
$\xi_4^D - \xi_{4_t}^D$	0.0156	0.0108	0.0450	0.0102

Table 26: Lamination parameters errors for the free vibration problem and for the buckling problem

It can be observed that the errors inversely reflect the weights introduced with the use of the strain energy. In fact, for the free vibration problem the errors are larger for the in-plane terms, while for the buckling problem the out-of-plate are the largest ones. For what concerns the errors on the lamination parameters linked to the in-plane laminate constitutive matrix and observing the orientation angles reported in Table 25, it is clear that with the objective function implemented, the optimization process cannot built a balanced laminate: the orientation angles are not in a "balanced" form and the errors on ξ_2^A and ξ_4^A are not zero.

The errors obtained from this conversion are comparable with the ones obtained in the validation process where the number of layers is not the correct one. This is due to the fact that the lamination parameters used in this conversion could not be represented by 16 layers with the same thickness. In fact, the lamination

parameters are obtained without knowing the exact configuration of layers and thicknesses.

The variable stiffness composite plate that are obtained with the conversions differs in the properties reported in Table 23, as shown in Table 27.

	Conversion Laminate	PSO outputted Laminate	% error
Free Vibration	11.83 Hz	11.87 Hz	0.3381
Buckling	2.25 N/mm	2.37 N/mm	5.0633

Table 27: Comparison between the conversion laminate and the "optimized" laminate

From Table 27, it can be observed that the percentual error on the first natural frequency is under the 0.5%, while for the first buckling load the error reaches the 7%. Starting from these data, it could be said that the difference between the laminate designed and the one that will be manufactured for the free vibration problem is negligible, but the same thing cannot be said for the buckling one.

7.7 Time consumptions of the process

The time spent for the process is the sum of all the time consumptions for the five different parts: the lamination parameters sampling, the extrapolation of the input-output training sets from a Ritz's analysis, the artificial neural network training phase, the particle swarm optimization phase, and the final conversion from the lamination parameters into orientation angles. All the time consumptions are reported in **Errore. L'origine riferimento non è stata trovata.**. The "TR (B → F)" row i s associated with the transfer-learning strategy, considering in each columns the sum of the time spent both for the buckling and for the free vibration problem.

	LP sampling [min]	Ritz's Analysis [min]	ANN Training [min]	PSO [min]	Post-processing [min]	TOTAL [min]
Free Vibration	35.94	75.69	27.27	1.66	11.41	151.97
Buckling	114.11	27.56	21.98	3.12	13.84	180.61
TR (B → F)	116.11	46.12	49.1	4.42	25.25	241

Table 28: Time spent in every process phase for the free vibration problem, for the buckling problem, and for the transfer-learning strategy(TR)

It can be observed that to solve the two example problems the transfer-learning strategy is preferable. In fact, a small increment of the level of error on the first natural frequency is obtained, but the time spent is largely reduced. With a standard procedure, the total time spent is about 332.58 minutes, while with a transfer-learning strategy is about 241 minutes. Therefore, using this latter strategy 91.58 minutes are saved. The part associated with the input-output pairs evaluation with the Ritz's analysis, and the lamination parameters sets sampling is the bottleneck of the process. In fact, most of the time spent is in these initial phases.

Chapter 8

Conclusions

In this work the optimization procedure using an artificial neural network and the particle swarm optimization algorithm was presented. The lamination parameters were implemented in the process in order to simplify the optimization procedures and reduce the number of the variables considered. This method is used to solve a free vibrational problem and a buckling problem for a rectangular simple supported variable stiffness plate, obtaining the maximum first natural frequency and the maximum first buckling load values. Finally, a method to obtain the orientation angles from the parameters used in the optimization was reported, allowing the manufacturability of the laminate.

In this analysis the errors are present in three different parts that are referred to the ANN accuracy, the PSO accuracy, and the conversion from lamination parameters to orientation angles. For what concerns the ANN, the maximum percentual errors are in the order of 3%, while the average percentual errors are in the range of 0.6% - 0.5%. This grants a good approximation for the values that are searched, with a minimum accuracy of 97%. The errors obtained from the results of the optimization process are in the order of 1%, that is in the prediction range (3%). From this level of error, it can be deduced that the penalty terms implemented did not introduce errors. The post-optimization phase produces a maximum error in the order of 0.2. The conversion modifies the properties of the plate, but the percentual errors between the laminate expressed with the lamination parameters and the one obtained with the conversion in orientation angles are under the 5%. Furthermore, the final laminates that can be manufactured violate the balanced assumption.

The level of error obtained both after the optimization process and after the post-processing process, demonstrate that this method is accurate enough and reliable to be implemented in design phase of composite structures.

The total time spent for the analysis is 332.58 minutes for the standard procedure (free vibration + buckling), and 241 minutes with a transfer-learning strategy (buckling + TR free vibration). If the ANN is substituted directly by a Ritz's analysis,

the post-processing time consumptions do not change. However, the lamination parameters sampling, the initial Ritz's analysis to determine the input-output pairs, and the ANN training are not involved in the process. The time consumption is only referred to the PSO. Therefore, if the dimension of the swarm is considered equal to the ones used in this work, and the PSO stops at the same number of iteration, the times spent for a PSO with a Ritz's analysis embedded are about 11880 minutes for the free vibration problem and 1440 minutes for the buckling one. However, the convergence can be reached before the number of iterations considered. Therefore, the number of iterations needed to spent the same time is evaluated, obtaining that for the free vibration problem the PSO has to converge in 36 iterations, while for what concerns the buckling problem this number is equal to 377.

From the data reported, it can be observed that with the same amount of iteration of the PSO, the method implemented in this work is better than the one that implements a Ritz's analysis inside the PSO. The number of iterations of this latter method for the free vibration problem is too small to grant the convergence, the one associated with the buckling problem, instead, could be enough to obtain the convergence. It has to be highlighted that over these numbers, the time consumption increases, in particular, for each iteration the time spent is equal to 4.32 minutes for the free vibration problem, and 0.48 minutes for the buckling one.

Furthermore, for more complex structures, the time saved could be larger. On the other hand, more lamination parameters sets have to be sampled and more time will be spent also on the evaluation of the input-output pairs. However, the sampled sets can be reused for problems that involved the same number of lamination parameters set to fully describe the structure.

From the data reported, both about errors and time consumption, the implemented method can be implemented, in particular the one that involved the transfer-learning strategy. In fact, with this training method, the time consumption is reduced significantly with a small increment on the error. Furthermore, the time spent for the entire process is less respect the one obtained with classical numerical analysis [6] or the one aforementioned.

8.1 Future innovations

The sample strategy used could be modified in order to improve the sampling distribution in the design space, trying to fill in a better way the space near the boundaries, and reducing the time consumption. The method used to obtain the input-output training sets could be substituted with other equivalent ones in order to check which one is the most performing. The Ritz's analysis can be substituted with FEM analysis or other types of numerical methods.

In the process could be considered different materials, that have to be insert in the input set in order to determine not only the more efficient fibres distribution, but also the better performing material in each layer.

More topologies of the ANN, different distributions of neurons through the layers, and other kind of training algorithms could be analysed to increase the performance of the surrogate model itself, aiming in an increment of the accuracy in a reduced amount of time. A single ANN with two or more different output could be built in order to compare the result obtained between that kind of network and the one used in this work.

Different optimization algorithms could be implemented to compare the different methods and find the more accurate one.

The thickness of each ply could be considered inside the post-processing process to increase the degrees of freedom. In this way it is allowed a better convergence between the lamination parameters and the ones relatives to the orientation angles obtained. Furthermore, other limitation could be introduced, i.e. the plies stacking sequence, the maximum angle difference between one ply and the following one, avoiding some configuration of angles for the more external plies, or the fibres distribution that voids some defected areas that would be possible location for failures [15].

References

[1] Honda S., and Narita Y. (2012). "Natural frequencies and vibration modes of laminated composite plates reinforced with arbitrary curvilinear fiber shape paths". Journal of Sound and Vibration, Vol.331, No.1, 180-191.

[2] Nik M.A., Fayazbakhsh K., Pasini D., and Lessard L. (2014). "A comparative study of metamodeling methods for the design optimization of variable stiffness composites". *Composite Structures*, Vol.107, 494-501.

[3] Setoodeh S., Abdalla M. M., and Gürdal Z. (2006). "Design of varible-stiffness laminates using lamination parameters". *ELSEVIER*, *Composite Structures*, Part B, Vol.37, 301–309.

[4] Setoodeh S., Blom A. W., Abdalla M. M., and Gürdal Z. (2006). "Generating curvilinear fiber paths from lamination parameters distribution". *AIAA* 2006-1875, *Structural dynamics*.

[5] IJsselmuiden S. T. (2011). "Optimal design of variable stiffness composite structures using lamination parameters". *ISBN*: 978-94-6113-032-7.

[6] Bisagni C., and Lanzi L. (2002). "Post-buckling optimisation of composite stiffened panels using neural networks". *Composite Structures*, Vol.58, 237-247.

[7] Oldani F. (2017). "Ottimizzazione di pannelli a rigidezza variabile attraverso il metodo PSO". *Thesis*.

[8] Liu W., and Butler R. (2004). "Buckling optimization of variable-angle-tow panels using the infinite-strip method". *Composite Structures*, Vol.63, No.1, 329–338.

[9] Gürdal Z., Tatting B. F., and Wu C. K. (2008). "Variable stiffness composite panels: Effects of stiffness variation on the in-plane and buckling response". *Composites* Structures, Part A: *Applied Science and Manufacturing*, Vol.39, No.5, 911–922.

[10] El-Sonbaty I., Khashaba U. A., and Machaly T. (2014). "Factors affecting the machinability of gfr/epoxy composites". *Composite Structures*, Vol.63, No.1, 329–338.

[11] Zheng Z., Bai R. X., Lei Z. K., Zhu Y. G., and Zhang X. Z. (2019). "Failure analysis of variable-stiffness laminate considering manufacturing defects". *IOP Publishing*, *The second Internation Workshop on Materials Science and Mechanical Engineering*. (DOI: 10.1088/1757-899X/504/1/012055).

[12] Diaconu C. G., Sato M., and Sekine H. (2002). "Layup optimization of symmetrically laminated thick plates for fundamental frequencies using lamination parameters". *Springer-Verlang*, *Structt Multidisc Optim,* Vol.24, 302-311. (DOI: 10.1007/s00158-002-0241-z).

[13] Liu X., Featherston C. A., and Kennedy D. (2019). "Two-level layup optimization of composite laminate using lamination parameters". *School of Engineering*, *Cardiff University*, *Queen's Buildings*, *The Parade*, Cardiff CF24 3AA, UK.

[14] Bloomfield M. W., Diaconu C. G., and Weaver P. M. (2008). "On feasible regions of lamination parameters for lay-up optimization of laminated composites". (DOI: 10.1098/rspa.2008.0380).

[15] Diaconu C. G., Sato M., and Sekine H. (2002). "Feasible region in general design space of lamination parameters for laminated composites". *AIAA Journal*, Vol.40, 559–565. (DOI: 10.2514/2.1683).

[16] Grenestedt J. L., and Gudmundson P. (1993). "Lay-up optimisation of composite material structures". *In Proc. IUTAM Symposium on Optimal Design with Advanced Materials*, 311–336.

[17] Herencia J. E., Weaver P. M., and Friswell M. I. (2007). "Optimization of long anisotropic laminated fiber composite panels with T-shaped stiffeners". *AIAA journal*, Vol.45, No.10, October, 2497-2509. (DOI: 10.2514/1.26321).

[18] Gürdal Z., Haftka R. T., and Hajela P. (1999). "Design and Optimization of Laminated Composite Materials". *Wiley-Interscience Publication*.

[19] IJsselmuiden S. T., Abdalla M. M., and Gürdal Z. (2008). "Implementation of strength-based failure criteria in the lamination parameters design space". *AIAA journal*, Vol.46, No.7, July, 1826-1834. (DOI: 10.2514/1.35565).

[20] Queipo N.V., Haftka R.T., Shyy W., Goel T., Vaidyanathan R., and Tucker P.K. (2005). "Surrogate-based analysis and optimization". *Progress in Aerospace Sciences*, Vol.41, 1-28.

[21] Vu K.K., D'Ambrosio C., Hamadi Y., and Liberti L. (2017). "Surrogate-based methods for black-box optimization". *International Transactions in Operational Research*, Vol.24, 393-424.

[22] Cardozo S.D., Gomes H.M., and Awruch A.M. (2011). "Optimization of laminated composite plates and shells using genetic algorithms, neural networks and finite elements". *Latin American Journal of Solids and Structures*, Vol.8, 413-427.

[23] Ruijter W., Spallino R., Warnet L., and de Boer A. (2003). "Optimization of composite panels using neural networks and genetic algorithms". *Second MIT Conference on Computational Fluid and Solid Mechanics*, Vol.179, 1-5.

[24] Shakeri M., Alibiglou A., and Abouhamze M. (2006). "Stacking Sequence Optimization of Laminated Cylindrical Panels Using a Genetic Algorithm and Neural Networks". *Conference Paper*, Vol.273, June, (DOI: 10.4203/ccp.83.273).

[25] Setoodeh S., Abdalla M. M., IJsselmuiden S. T., and Gürdal Z. (2009). "Design of variable-stiffness composite panels for maximum buckling load". *Composite Structures*, Vol.87, No.1, 109–117.

[26] Liang Z., and Yupu Y. (2009). "PSO-based single multiplicative neuron model for time series prediction". *Expert System with Application*, Vol.36, 2805-2812.

[27] Aladag C.H., Yolcu U., and Egrioglu E. (2013). "A new Multiplicative Seasonal Neural Network Model Based on Particle Swarm Optimization". *Neural Process Lett*, Vol.37, 251-262.

[28] Ramachandran P., Zoph B., and Le Q.V. (2017). “Searching for Activation Functions”. *Google Brain Residency program*, October, arXiv 1710.05941v2.

[29] Santos R.B., Rupp M., Bonzi S.J., and Fileti A.M.F. (2013). “Comparison Between Multilayer Feedforward Neural Networks and a Radial Basis Function Network to Detect and Locate Leaks in Pipelines Transporting Gas”. *Chemical Engineering transactions*, Vol.32.

[30] Jin Y., Li J., Du W., and Qian F. (2016). “Adaptive Sampling for Surrogate Modelling with Artificial Neural Network and Its Application in an Industrial Cracking Furnace”. *The Canadian Journal of Chemical Engineering*, Vol.94, February, 262-272.

[31] Ong Y.S., Nair P.B., and Keane A.J. (2003). “Evolutionary Optimization of Computationally Expensive Problems via Surrogate Modeling”. *AIAA Journal,* Vol.41, No.4, April.

[32] Papadopoulos V., Soimiris G., Giovanis D.G., and Papadrakakis M. (2018). “A neural network-based surrogate model for carbon nanotubes with geometric nonlinearities”. *Comput. Methods Appl. Mech. Engrg*., Vol.328, 411-430.

[33] Wilamowski B.M. (2009). “Neural Network Architectures and Learning Algorithms – How Not to Be Frustrated with Neural Networks”. *IEEE Industrial Electronics Magazine*, December, 56-63. (DOI: 10.1109/MIE.2009.934790).

[34] Bishop C. M. (2006). "Pattern recognition and machine learning". *Springer Science+Business Media*, *LLC*.

[35] Yu H., Xie T., and Wilamowski B.M. (2011). “Advantages of Radial Basis Function Networks for Dynamic System Design”. *IEEE Transactions on Industrial Electronics*, Vol.58, No.12, December, 5438-5450.

[36] Xie T., Yu H., and Wilamowski B.M. (2011). “Comparison between traditional neural networks and radial basis function networks”. *IEEE, Conference Paper*, July. (DOI: 10.1109/ISIE.2011.5984328).

[37] Beigy H., and Meybodi M.R. (2009). “A Learning Automata Based Algorithm for Determination of the Number of Hidden Units for Three

Layers Neural Networks". *International Journal of Systems Science*, January. (DOI: 10.1080/00207720802145924).

[38] Karlik B., and Olgac V. (2010). "Performance Analysis of Various Activation Functions in Generalized MLP Architectures of Neural Networks". *International Journal of Artificial Intelligence and Expert Systems*, Vol.1, No.4, 111-122.

[39] Zhang C., and Woodland P.C. (2015). "Parameterised Sigmoid and ReLU Hidden Activation Functions for DNN Acoustic Modelling*". ISCA, Natural Speech Technology, Dresden, Germany, September*, 3224-3228.

[40] Kathirvalavakumar T., and Subavathi S.J. (2009). "Neighbourhood based modified backpropagation algorithm using adaptive learning parameters for training feedforward neural networks". *Neurocomputing*, Vol.72, 3915-3921.

[41] Subavathi S.J., and Kathirvalavakumar T. (2011). "Adaptive modified backpropagation algorithm based on differential errors". *International Journal of Computer Science, Engineering and Applications*, Vol.1, No.5, October.

[42] Ranganathan V., and Natarajan S. (2018). "A New Backpropagation Algorithm without Gradient Descent". arXiv 1802.00027v1.

[43] Wilamowski B.M., Cotton N.J., Kaynak O., and Dündar G. (2008). "Computing Gradient Vector and Jacobian Matrix in Arbitrarily Connected Neural Networks". *IEEE Transactions on Industrial Electronics*, Vol.55, No.10, October, 3784-2790.

[44] Wilamowski B.M., and Yu H. (2010). "Neural Network Learning without Backpropagation". *IEEE Transactions on Neural Networks*, Vol.21, No.11, November, 1793-1803.

[45] Xie T., Yu H., Hewlett J., Różycki P., and Wilamowski B.M. (2012). "Fast and Efficient Second-Order Method for Training Radial Basis Function Networks". *IEEE Transactions on Neural Networks and Learning Systems*, Vol.23, No.4, April, 609-619.

[46] Venter G., and Sobieski J.S. (2003). "Particle Swarm Optimization". *AIAA Journal*, Vol. 41, No. 8, August.

[47] Ferreira Cruz D.P., Maia R.D., da Silva L.A., and de Castro L.N. (2016). “BeeRBF: A bee-inspired data clustering approach to design RBF neural network classifiers”. *Neurocomputing*, Vol.172, 427-437.

[48] Qasem S.N., Shamsuddin S.M., and Zain A.M. (2012). “Multi-objective hybrid evolutionary algorithms for radial basis function neural network design”. *Knowledge-Based Systems*, Vol.27, 475-497.

[49] Aladag C.H. (2011). “A new architecture selection method based on tabu search for artificial neural networks*”. Expert System with Applications*, Vol.38, 3287-3293.

[50] Ghashochi Bargh H., and Sadr M. H. (2012). "Stacking sequence optimization of composite plates for maximum fundamental frequency using particle swarm optimization algorithm". Meccanica, Vol.47, 719-730. (DOI: 10.1007/s11012-011-9482-5).

[51] Tse P. C., and Lai T. C. (1994). "An expression for the strain energy of laminated composite thin shells". *International Journal of Mechanical Engineering Education*, Vol.23, No.2, 169-177.

[52] Vescovini R., Oliveri V., Pizzi D., Dozio L., and Weaver P. M. (2019). "A semi-analytical approach for the analysis of variable-stiffness panels with curvilinear stiffeners". International Journal of Solids and Structures, https://doi.org/10.1016/j.ijsolstr.2019.10.011.

[53] Crawley E. F. (1979). " The natural mode shapes and frequencies of graphite/epoxy cantilever plates and shells". *Journal of composite materials*, Vol.13, 195-205.

[54] Wu Z., Weaver P. M., Raju G., and Kim B. C. (2012). "Buckling analysis and optimisation of variable angle tow composite plates". Thin-Walled Struct, Vol.60, 163–172.

[55] Bisagni C., Vescovini R., and Dávila C. G. (2011). "Single-stringer compression specimen for the assessment of damage tolerance of postbuckled structures". *Journal of Aircraft*, Vol.48, No.2, March-April, 495-502, (DOI: 10.2514/1.C031106).