DACOMAT

DELIVERABLE NUMBER: 5.1

Advanced algorithms for damage detection from strain measurements

Alfredo Güemes, Antonio Fernandez-Lopez, Daniel del Rio, José Garcia-Blanco, Christian Aguilar

Universidad Politecnica de Madrid, Spain

The DACOMAT project has received funding from the European Union's Horizon 2020 research and innovation programme under GA No. 761072

Deliverable details

Document name: **Advanced Algorithms for damage detection from strain measurements**
Responsible partner: UPM, Spain
Work package: 5
Task: 5.1
Due date: 30/12/2019
Delivery date: 30/12/2019
Author(s): A. Güemes, A. Garcia-Lopez, D. del Rio, J. Garcia Blanco, C. Aguilar

Revision history

Version	Date	Author / Reviewer	Notes
1	01/12/2019	A. Güemes	
	15/12/2019	M. McGugan	
	22/12/2019	J.K.Jørgensen	

Dissemination level	
X	PU = Public
	CO = Confidential, only for members of the consortium (including the EC)
	Classified, information as referred to in Commission Decision 2001/844/EC.

Deliverable type	
X	R: Document, report (excluding the periodic and final reports)
	DEM: Demonstrator, pilot, prototype, plan designs
	DEC: Websites, patents filing, press & media actions, videos, etc.
	OTHER: Software, technical diagram, etc.

The DACOMAT project has received funding from the European Union's Horizon 2020 research and innovation programme under GA No. 761072

Popular science summary (Public)

Structural Health Monitoring (SHM) is defined as the process of acquiring and analysing data from on-board sensors to evaluate the health of the structures. A technology still under development, it offers significant savings for the maintenance of structures, by replacing the current time-scheduled inspections, by condition-maintenance inspections, only when needed. Fibre-optic sensors (FOS) are very attractive as sensors for SHM applications; its size is very small, the optical fibre has a diameter of 150 microns, so it can be embedded within the composite material during manufacturing. Without doubt they are the best technical choice for strain measurements. But damage occurrence at the structure cannot be directly derived from strain measurements; local changes are intense at the crack tip, but smooth out very quickly. Global changes in the strain field caused by a small damage, such as a crack, delaminations or debondings are quite small, easily hidden by the equipment noise and environmental disturbances. Several local and global SHM techniques have been proposed, in order to derive damage information from strain measurements. Local SHM techniques are more sensitive, and better developed, but they are restricted to damages located at the area covered by the sensors. This paper deals with global methods, which are supposed to detect the damage anywhere in the structure. Two methods are proposed, and the concept is demonstrated on a bridge beam.

Technical summary

The goal of this task has been to develop algorithms to compare the strain data obtained on a pristine structure, with the data flow generated during the service life of the structure. The most classical algorithm is PCA (Principal Component Analysis). It is a robust algorithm that does not require any hypothesis about the damage type, and produce for each new data set a 'Damage Index', or deviation among the new data and those from the pristine structure. During this project, refinements have been introduced to improve the sensitivity of the technique to small damages. Another algorithm, based on Recurrent Neural Networks (RNN) has been developed and proven with the data simulated for the bridge beam. The performances of the NN are strongly linked to the quality of data, both from the pristine and from the damaged structures; when the number of simulated cases is adequately large, the predictions are quite accurate, with a capability to identify and to classify the damages. It is worthy to clarify that both algorithms are included among the Data-driven group; they do not require any knowledge of the physics of the system. Nevertheless, a physical model is needed for the simulation, based on Finite element Modelling (FEM), to produce the strain data at the assumed positions of the sensors. It allow an optimization of the sensor network, and a verification of the algorithms without the need to run real experiments.

The DACOMAT project has received funding from the European Union's Horizon 2020 research and innovation programme under GA No. 761072

Contents

List of Figures

List of Tables

Introduction

Damage is a local change in the material's properties or at the structure boundaries that degrades structural performance. This aspect cannot be directly measured, so damage detection requires the measurement of a physical parameter affected by damage presence such natural frequencies. Although the changes induced by damage appearance in the strain field are obvious (intense in the local field, and smaller in the global strain map), damage detection and location by a sparse sensor strain network requires a damage detection algorithm which provide an accurate detection of damage presence, location and quantification.

Among the strain sensors, Fiber Optic Sensors (FOS) are very attractive as sensors for SHM applications on aerospace or civil composite structures, because optical fibers have very small size, low weight and multiplexing capabilities (several sensors on the same optical fibre). Additionally, they can be embedded within the composite material during manufacturing. Other benefits of FOS are EMI/RFI immunity, long term stability, wide temperature range and very long cabling if needed because of the low attenuation.

Some damage detection algorithms have been studied using simulation with a Finite Element Model (FEM) of a structure, particularly, a beam. In this model, some element rows are used as distributed measurement FOS, with which the deformations needed to perform the damage detection algorithms have been extracted.

1.1 SHM methods based on strain measurements

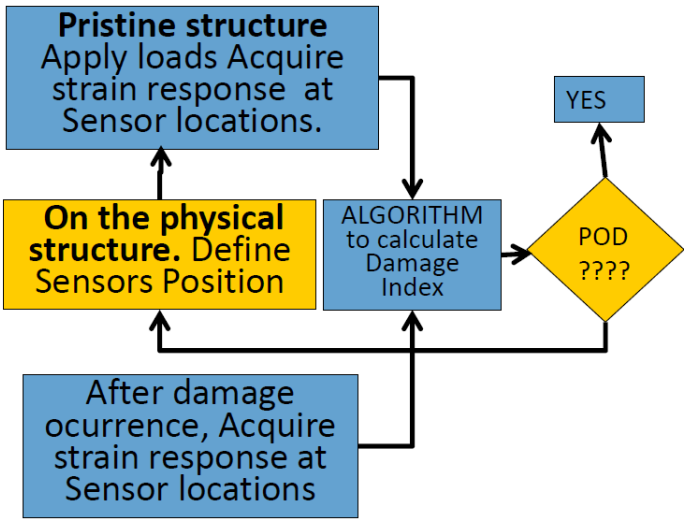
Strain measurements are very popular for damage detection strategies. Due to the limited spatial influence of the changes induced in the strain field, several applications are based on positioning the sensor in the expected damage area, like the bonding line between the stringer and the skin on composite structures. In the case of distributed sensing networks, larger measuring areas are possible. Although strain sensors have high maturity, a key issue to determine the damage straight from strain measurements is to compensate for the unknown influence of external loads and temperature on the strain field. This, together with the limited sensing area, are currently the major limitations to implement this technique on field applications.

Strain sensors are commonly used in vibration-based methods, which inherently have the advantage of a global survey of the whole structure, and the limitation that damage needs to be large enough to modify the modal shapes. Other widespread algorithms are Principal Component Analysis (PCA). PCA is a simple and nonparametric method of extracting relevant information directly from strain data. It is done by reducing a complex dataset to a lower dimension, revealing some hidden structure/patterns or abnormal data. This is done by converting a set of data of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components. In the same line, Support Vector Machines (SVMs) classification learning is a powerful paradigm to investigate the inverse input-output relationship of a specific problem according to some available and representative dataset.

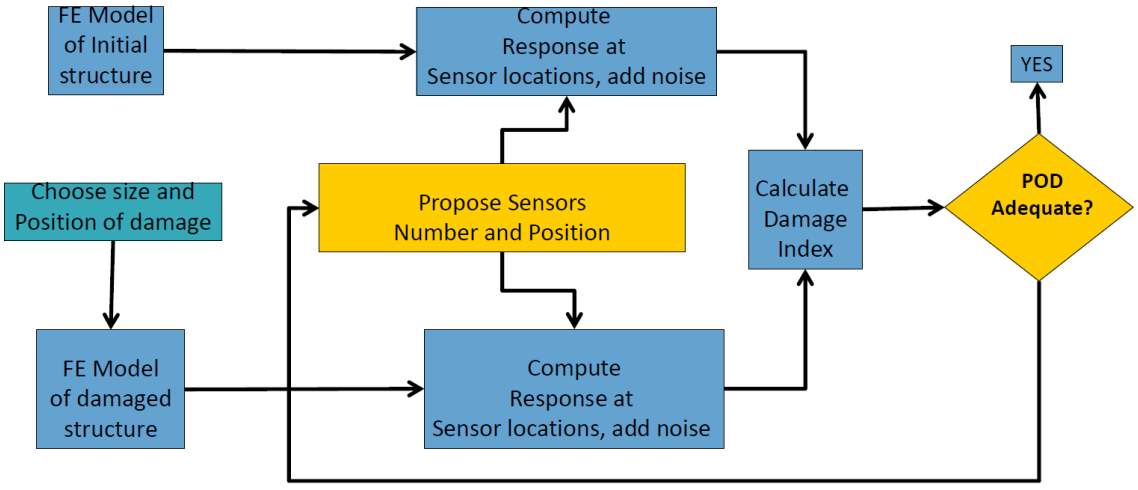
Due to its intrinsic nature, machine learning algorithms match perfectly with SHM applications, as

they are designed to find natural patterns in data that generate insight. Different applications can be found in the bibliography. Initially, Artificial Neural Networks (ANN) were used to determine changes in vibration or in time series. However, new developments in Recurrent Neural Networks (RNN) provide the capability to detect damage presence based on the structure strain field. Fatigue cracks and the distortion of the FBG spectrum have previously been studied.

The following Figure 1.1.1 shows the block diagram used for a real case or when you have a model.



(a) Algorithm to process experimental data



(b) Algorithm for a strain-based SHM system to evaluate multiple damage scenarios

Figure 1.1.1: Algorithms block diagram

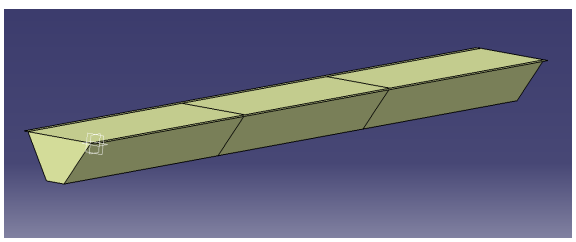
Structure under study

2.1 Geometry

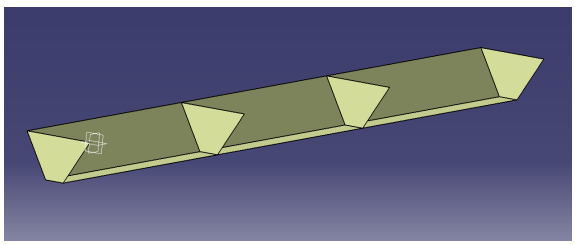
This study had been perfomed using a structure made of a beam. This beam has a constant section with an Omega shape, in which lateral panels form a 60º angle with the top panel. The beam has a total length of 10 meters and a height of 0.5 meters in between the first and last rib and top and bottom surfaces respectively.

Figure 2.1.1: Example beam

The first rib is placed at the beginning (z = 0), the second at z = 3400 mm (with respect to the first rib), the third at 6600 mm and the forth and last at the end of the beam, 10000 mm.



(a) External surfaces



(b) Ribs surfaces

Figure 2.1.2: CAD design

2.2 Material

The structure is made of two different GRP laminates. The top and bottom surfaces have a 10 mm thick laminate while the laterals and ribs are 4mm.

The complete information on the laminates can be seen in table 2.2.1, where the thickness values are in mm and the orientation in Degrees.

Laminate thickness	Layer thickness	Number of layers	Orientation
10	0.2	50	$[(\pm 45, 90, 0, 90), 0_{15}]_S$
4	0.2	20	$[(\pm 45, 90, 0_7]_S$

Table 2.2.1: Laminates

The 0º direction matches the longitudinal direction of the beam.

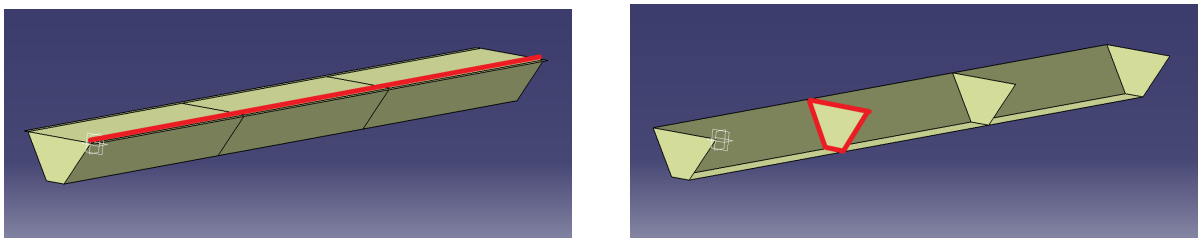
2.3 Damages to detect

The main objective of this study is to prove the ability of algorithms to detect damage presence in structures, therefore, the damages have to be defined. They are the partial peel off of the top surface and for the first rib.

Ultimately, the algorithm will have to classify a strain field between three states:

1. Top surface debond (D1)
2. Second rib debond (D2)
3. Undamaged structure (Ud)

The damages are made by a progressive bonding, starting with a total debonding of the surface and increasing in 20% of the surface on each step until 20% of the surface remains unpeeled.



(a) Top surface debond: Damage 1 (b) Second rib debond: Damage 2

Figure 2.3.1: Damage location

2.4 Finite Elements Model

Once the geometry is defined it can be exported to a FEM software, in this case, to Patran. With this method a big problem is reduced into many simple problems in which K matrix is easily computable and Nastran, the post processor, finally obtains the strains.

The method will be explained below:

2.4.1 Geometry

The geometry can be imported from a CAD model or doing the surface from scratch by introducing all points coordinates, create the curves joining points and the surfaces connecting the curves. In this study the last method was used.

2.4.2 Meshing

Preparing a good mesh is one of the most important parts of the process. This structure can be considered as a simple one due to the lack of curvature changes, thicknesses variation inside surfaces or mass discontinuities, therefore no dense meshing is needed.

The beam has 4 different surfaces which can be seen in Figure 2.4.1. The mesh properties at each surface are displayed in the tables below:

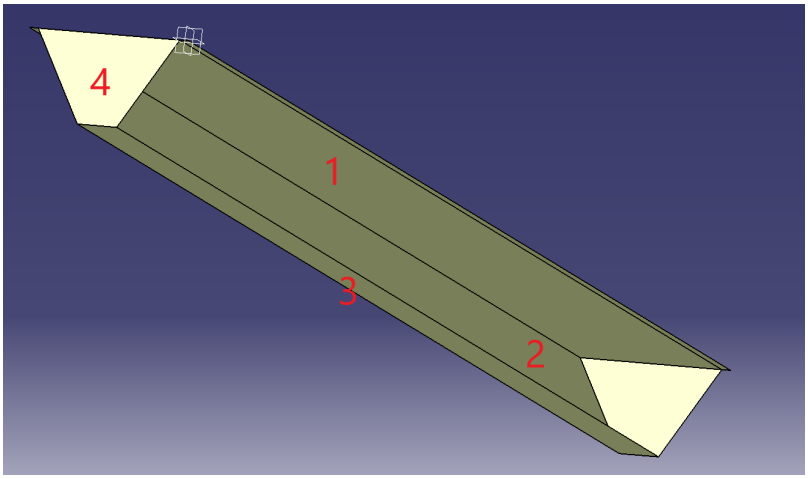


Figure 2.4.1: Numbered surfaces

The beam has 4 different surfaces and the repeated ones are meshed in the same way.

The meshing has been done by dividing the corresponding curve into an integer number of elements of equal size. This means that each surface has $N_i \cdot N_j$ elements and $(N_i + 1) \cdot (N_j + 1)$ nodes. The properties at each surface are displayed in Table 2.4.1:

Surface	Curves	Elements (1/2-Total)	Nodes
1	Longitudinal/Top Cross	50/17 - 850	51/18 - 918
2	Longitudinal/Lateral vertical	50/12 - 600	51/13 - 663
3	Longitudinal/Bottom cross	50/5 - 250	51/6 - 306
4 (irregular)	Top Cross/Lateral vertical/Bottom cross	17/12/5	18/12/5

Table 2.4.1: Top and bottom laminate

The model information of the undamaged state is shown in Figure 2.4.2.

 The DACOMAT project has received funding from the European Union's Horizon 2020 research and innovation programme under GA No. 761072

M O D E L	S U M M A R Y	BULK = 0
ENTRY NAME	NUMBER OF ENTRIES	
CQUAD4	3120	
FORCE	32	
GRID	3082	
LOAD	1	
MAT1	1	
MDLPRM	1	
PARAM	2	
PSHELL	2	
SPC1	1	
SPCADD	1	

Figure 2.4.2: Model information

2.4.3 Loads and Boundary Conditions

The forces in a FEM are transmitted through the nodes, so the loads and the boundary conditions have to be applied at them.

- The BC are applied at the lower nodes of the ribs at the ends of the beam. It consist of fixed positioning on the three axes.
- The force is applied to the upper nodes of the central ribs, with a magnitude of $32x10^7$ N distributed for each node (16 at each rib, 32 altogether) and with the negative direction of the axis Y.

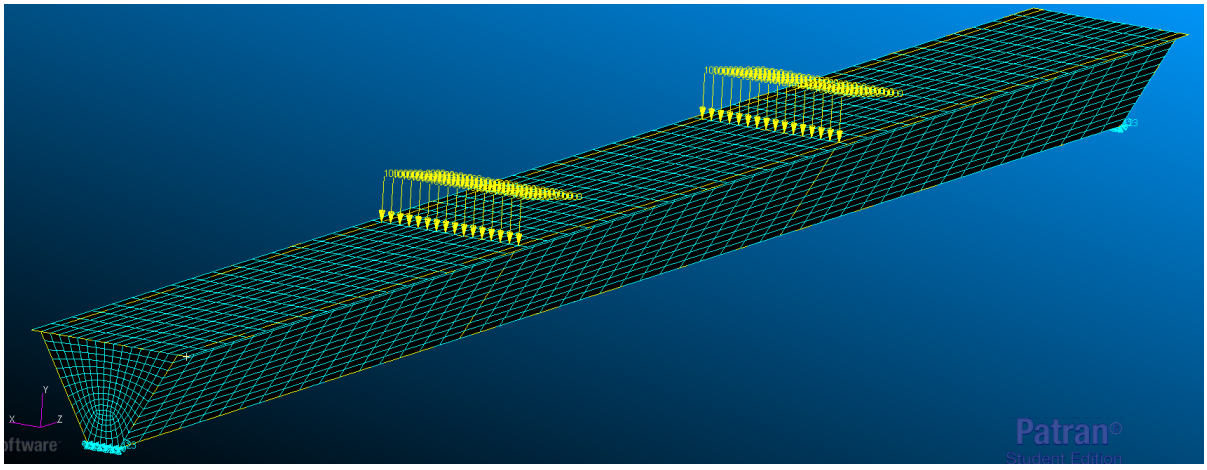


Figure 2.4.3: Load and BC applied

2.5 Results

Once the FEM is completed, we proceed to analyse the various structure states. After the post processsing is done, the desired element strains are extracted to feed the damage detection algorithms.

As previously stated, there are two damages which are made in a progressive way. In a FEM model, when you mesh two surfaces with a common curve, each surface has their own nodes belonging to the

curve. In other words, the surfaces are independent unless the nodes are joined. Therefore, the way to produce the debond was to not join the nodes of the surfaces involved.

We start with a complete debond (no joined nodes) and, at each damage step, the correspondent nodes have been linked as can be seen in Figure 2.5.1.

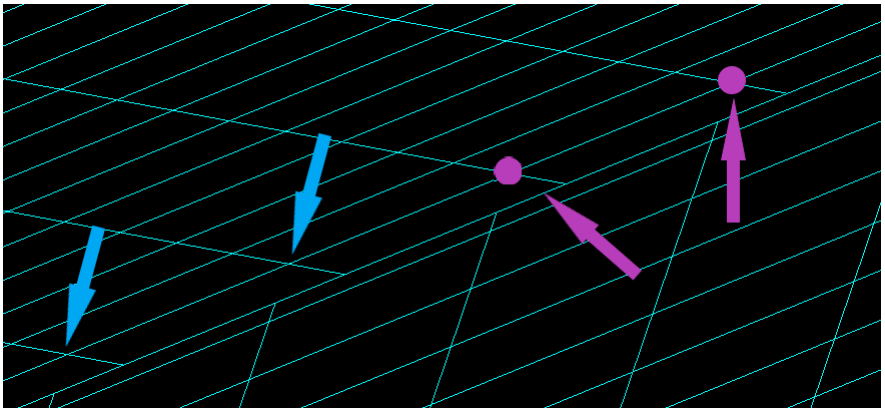


Figure 2.5.1: Joined (blue arrow) and unjoined (pink) nodes

When a damage is made, the next step is to analyze the model and introduce the data in Nastran. After postprocessing the results with Nastran and introducing back into Patran, the deformations can be plotted. An example is shown at Figure 2.5.2

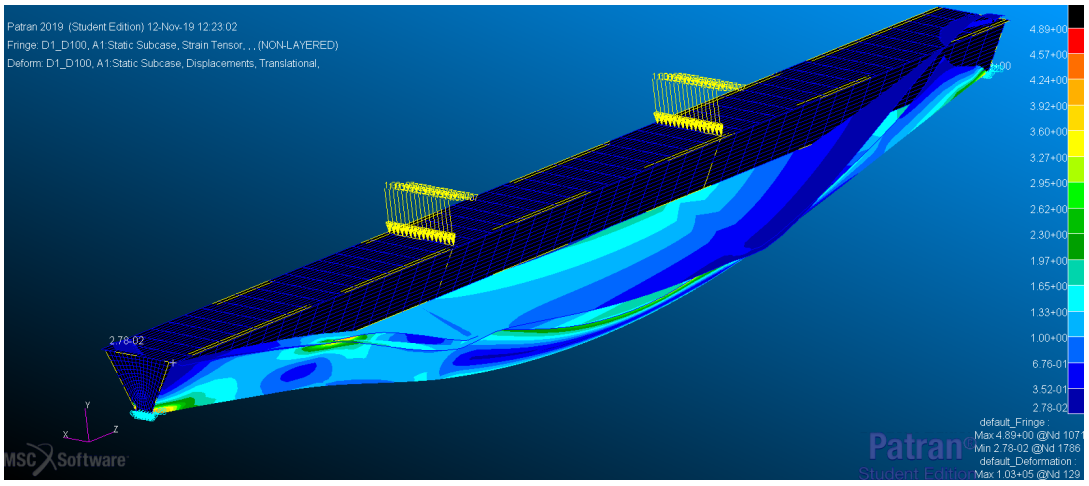


Figure 2.5.2: Damage 1 with a 80% of the total debond

The damage detection algorithm cannot be fed with the whole strain field which is extracted from the FEM model, because in a real structure the instrumentation of the complete structure is not possible. So, nine rows of elements have been selected to simulate a distributed FOS net. This sensor net can be installed in a real structure without problems.

 The DACOMAT project has received funding from the European Union's Horizon 2020 research and innovation programme under GA No. 761072

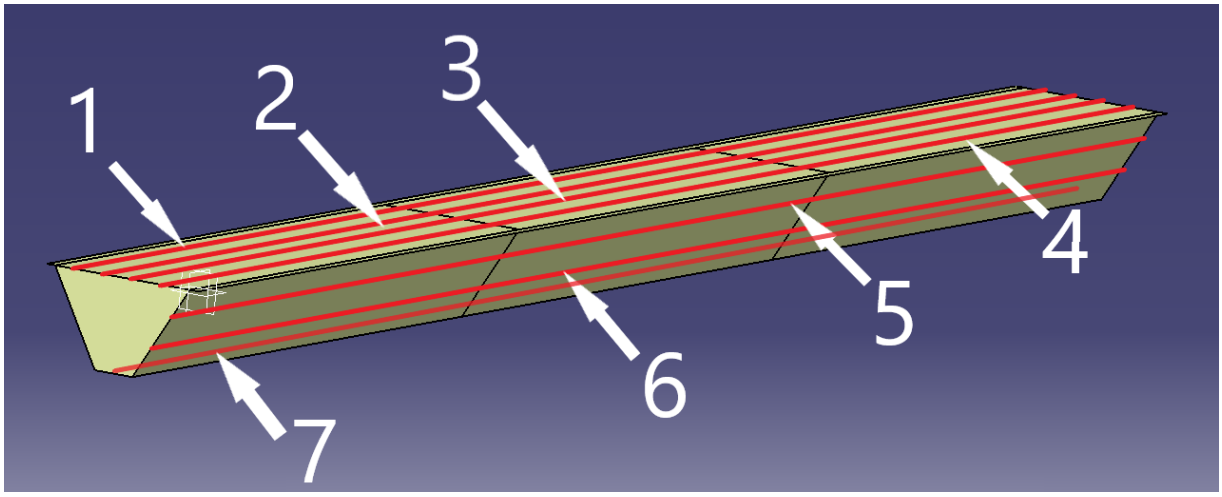


Figure 2.5.3: Distributed FOS

The distributed FOS can only measure the deformation of the structure in one direction. For this reason it's important to know the orientation of the local coordinate system of the elements due to the fact that we need to extract the deformation value of the element axis that matches with the longitudinal direction at the sensor.

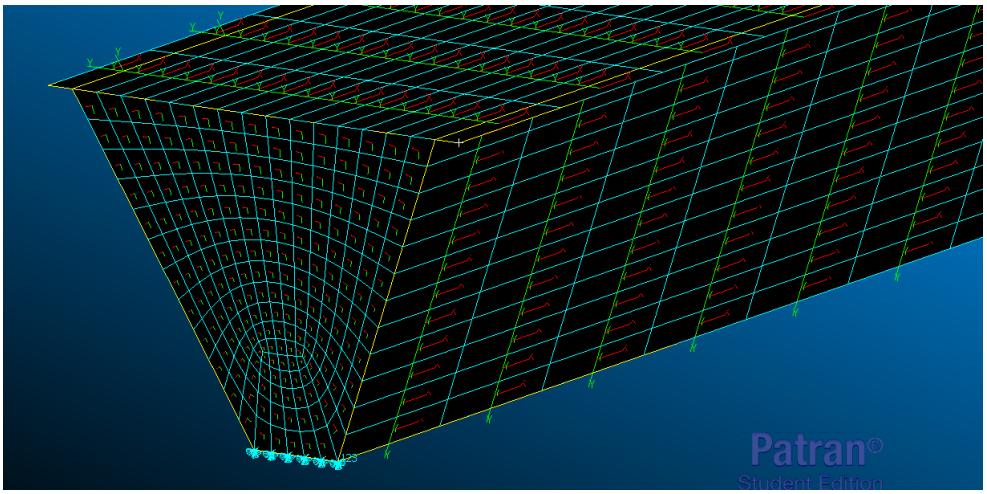
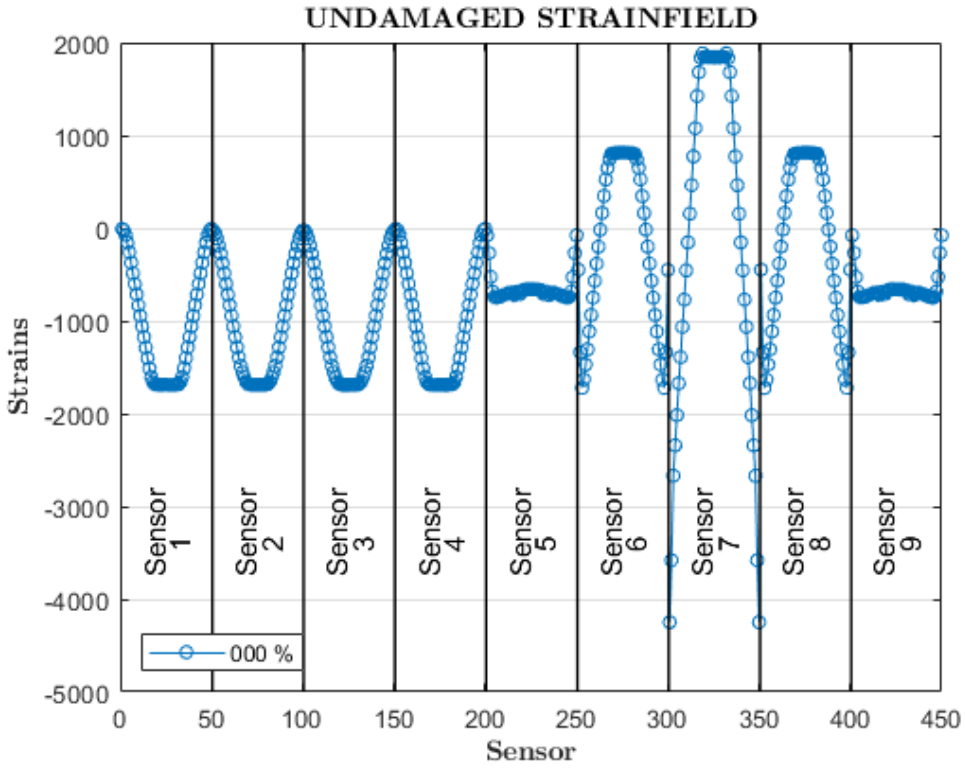


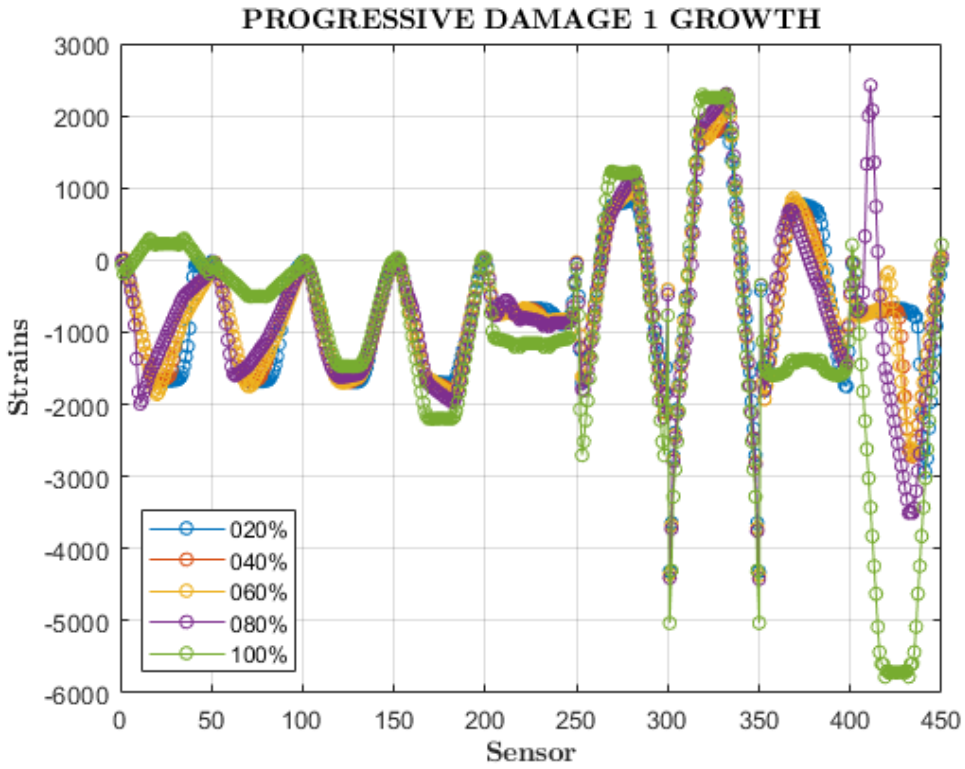
Figure 2.5.4: Elements's local coordinate system

Fortunately all the elements, except the ribs' ones, have their X-axis aligned with the sensor direction. This makes the data extraction easier.

The element strains (in the defined direction) are stored in a .rpt file that is generated by Patran, nevertheless, this file contains all the elements of the structure and the algorithms that are going to be fed with the FOS net elements. So, a Matlab code has been created in order to extract the strains belonging to the net. This information is plotted in Figure 2.5.5 for all the structure states.

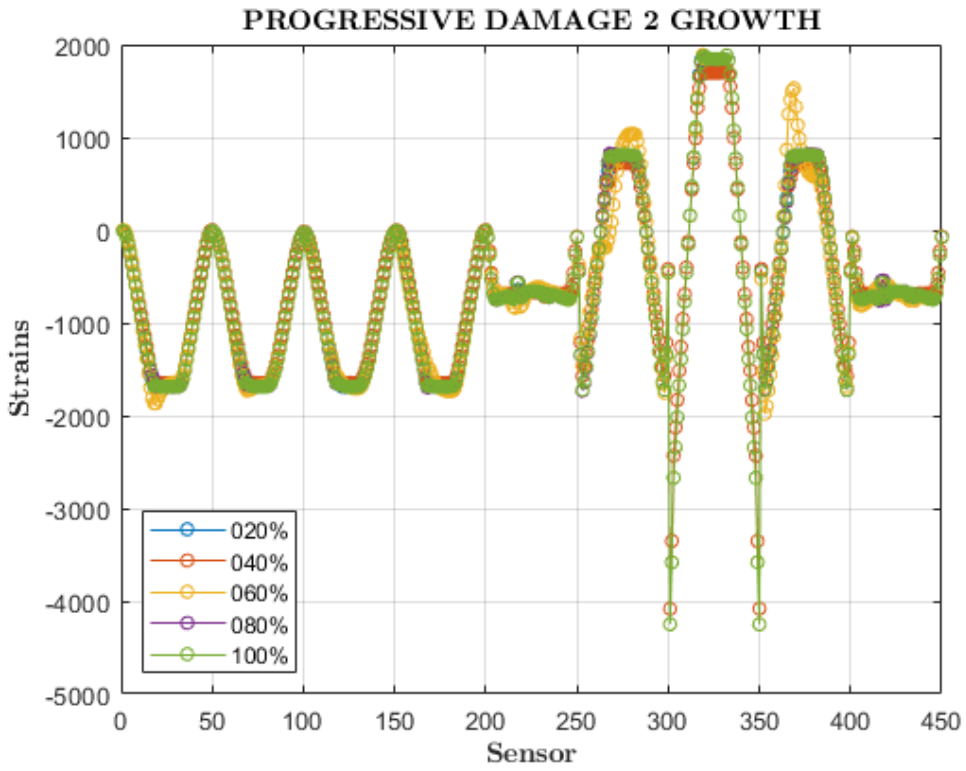


(a) Undamage state



(b) Damage 1: top surface dobonding

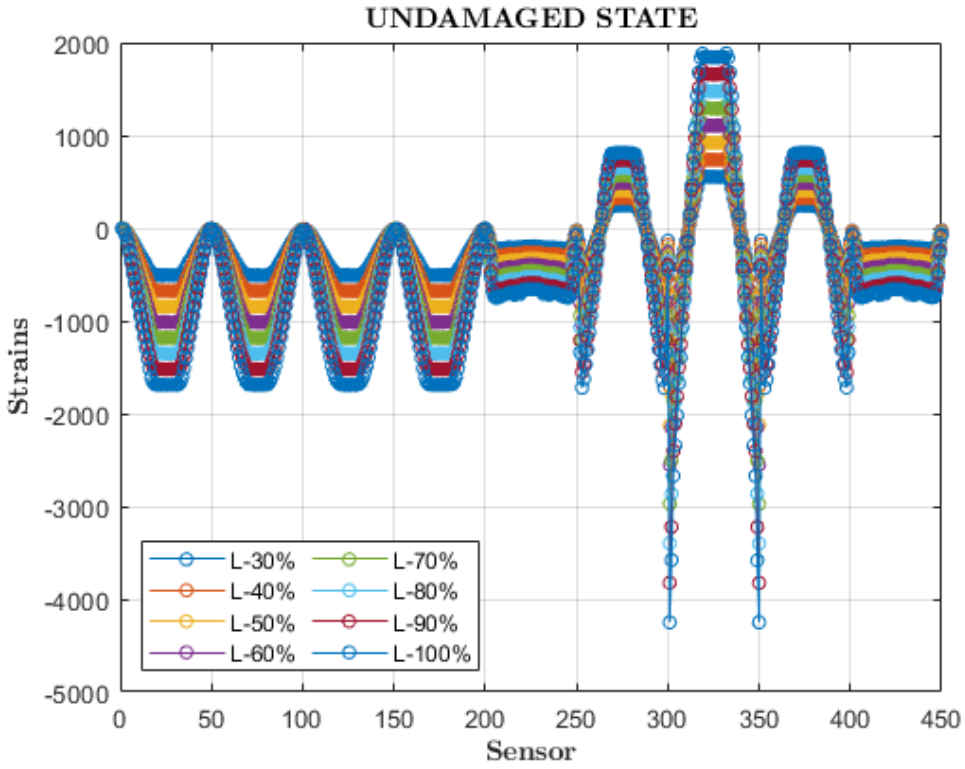
 The DACOMAT project has received funding from the European Union's Horizon 2020 research and innovation programme under GA No. 761072



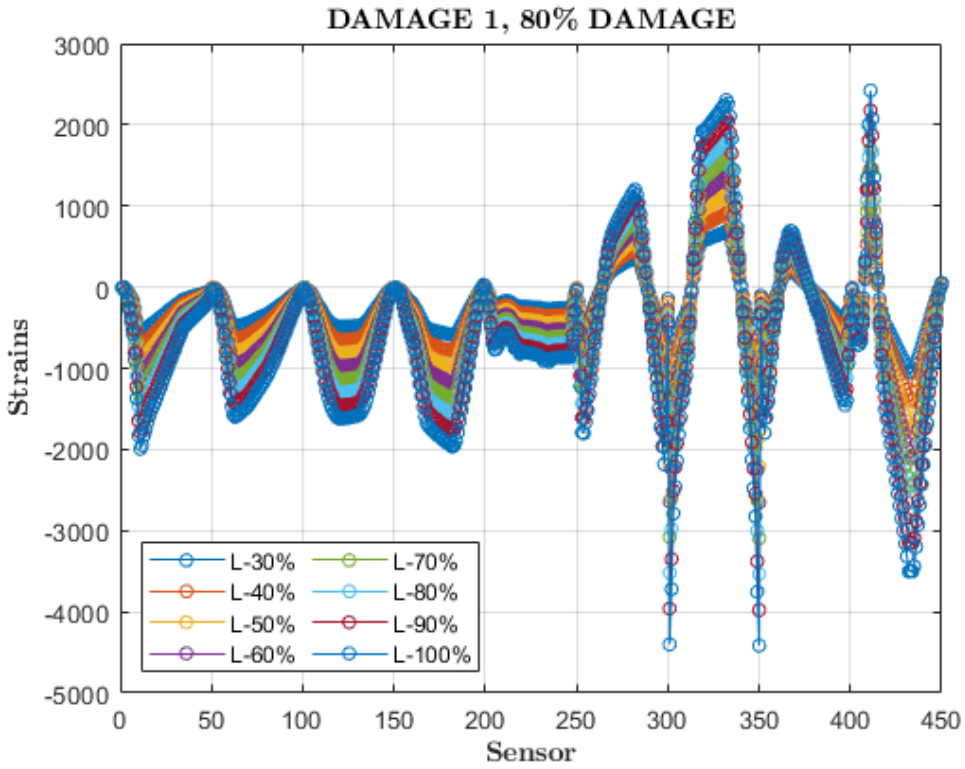
(c) Damage 2: rib dobonding

Figure 2.5.5: Sensor net strainfields exstracted from .rpt file

Nevertheless, the damage detection algorithms cannot be fed only with just the strains generated by one single value load since a structure can be subjected to different loads. It has to be remembered that the strains have a linear dependence with the load, so that, if a force with a value of 50% to the maximum load is applied, the strain will be a half of maximum. Therefore, the strains at each damage have been scaled from 100% to 30% in 10% steps, in this way each structure state has 8 load cases. This result can be seen in Figure 2.5.6

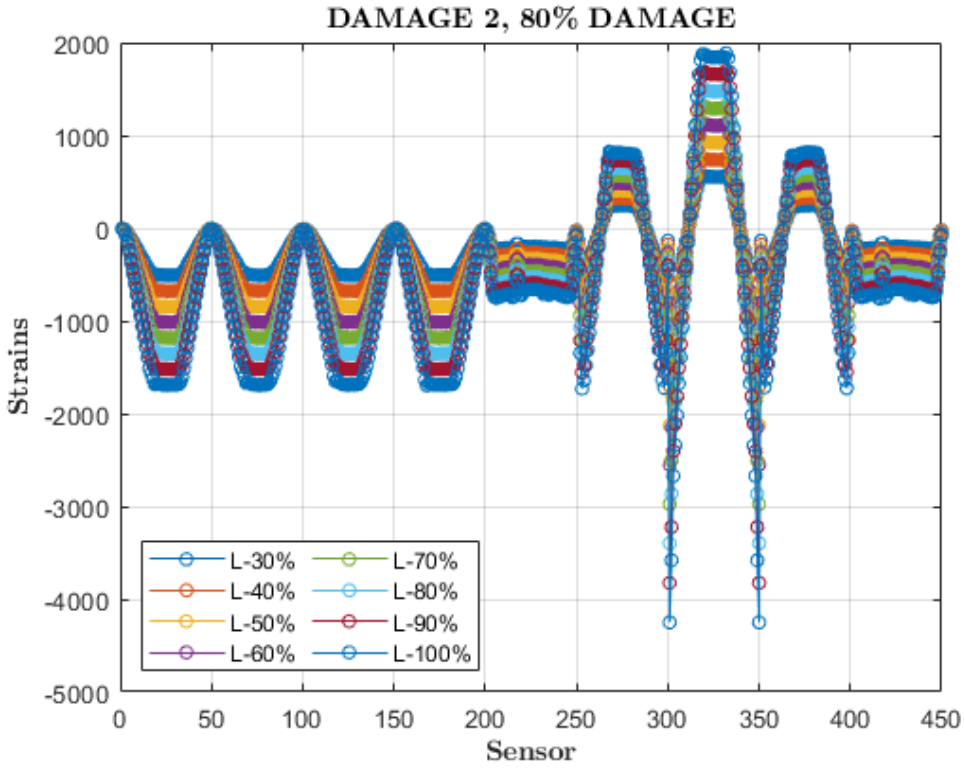


(a) Undamage state



(b) Damage 1, 80% of damage

 The DACOMAT project has received funding from the European Union's Horizon 2020 research and innovation programme under GA No. 761072



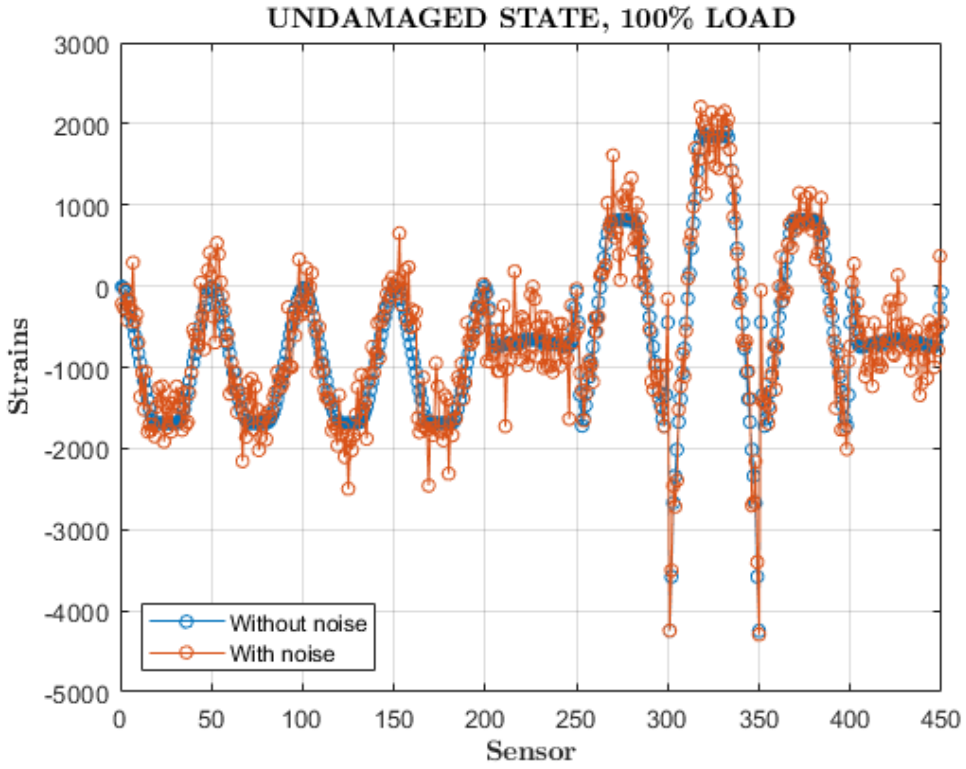
(c) Damage 2, 80% of damage

Figure 2.5.6: Strains provoked by a scaled load

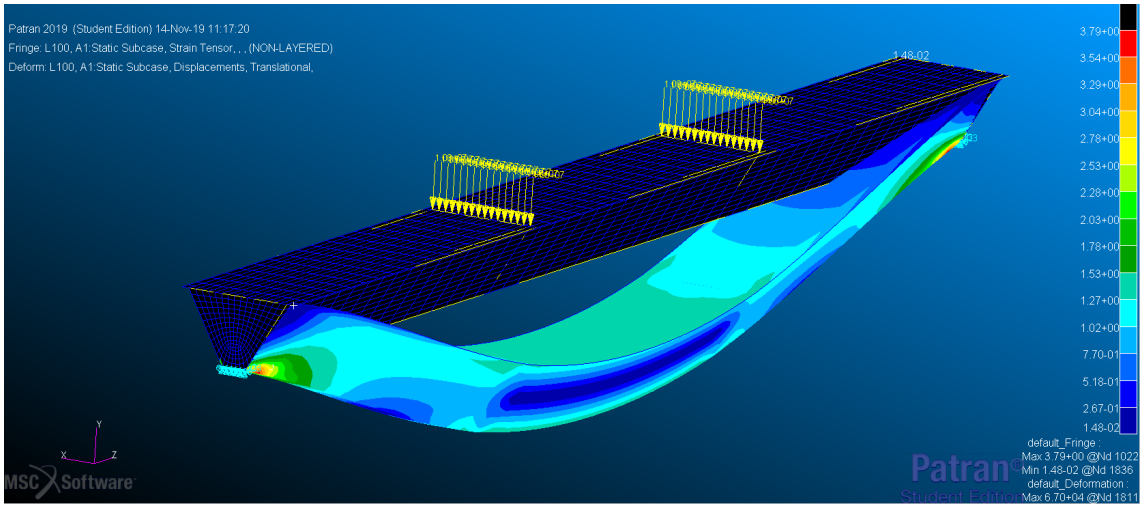
At this point, we have the deformations on an ideal structure. However, if FOS sensors are used to read the strain field in a structure in a real test, noise will come up with the signal received from them. Therefore, a Gaussian noise is added with the Matlab code to simulate this noise.

The maximum amplitude of the noise in these sensors is about 5 microstrains, which is the value that has been added at the time of carrying out the study. However, as can clearly be seen in the example in Figures 2.5.8, 2.5.9 and 2.5.7 noise has been added with an amplitude of 50. The Patran 3D plots are included in order to have a general idea of the deformation state of the structure at each state.

The DACOMAT project has received funding from the European Union's Horizon 2020 research and innovation programme under GA No. 761072



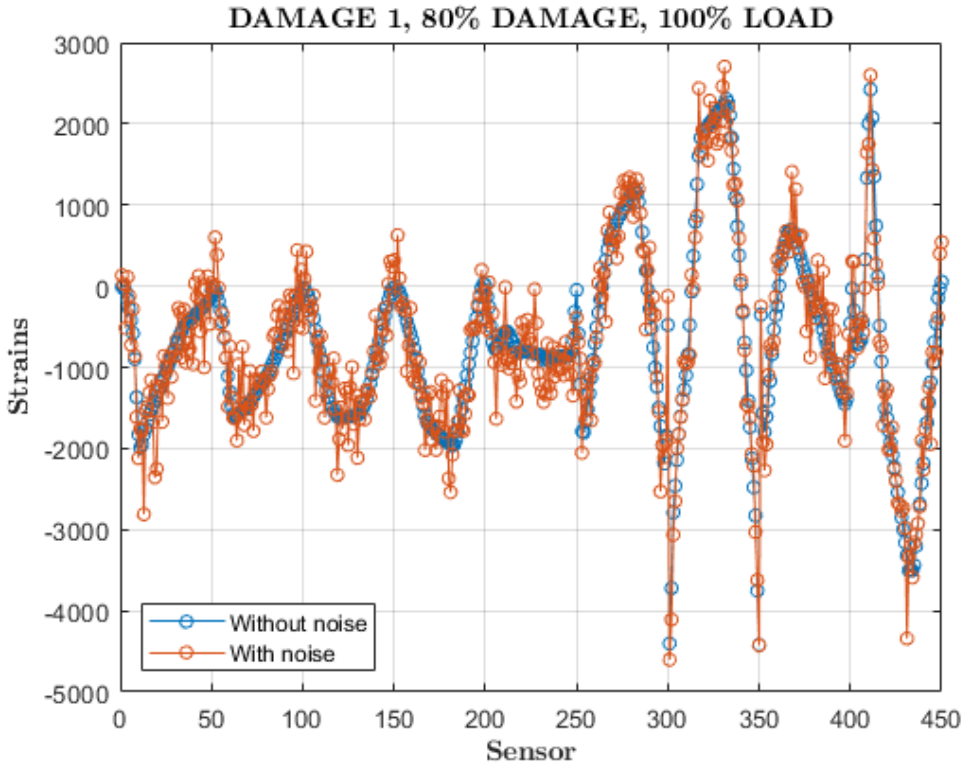
(a) Noise comparison at Undamaged state



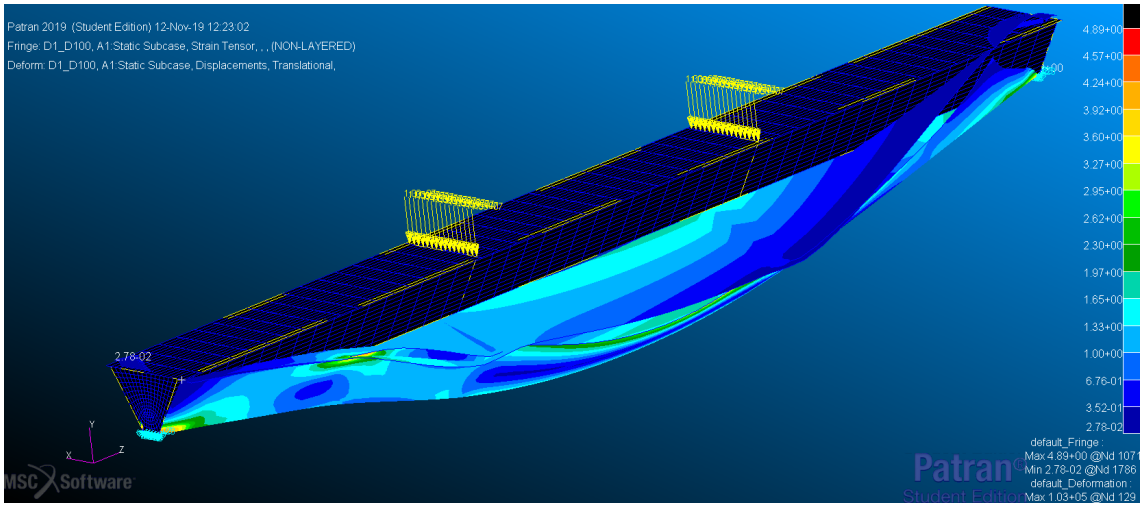
(b) Patran deformation plot

Figure 2.5.7: Undamaged state noise comparison and 3D plot

 The DACOMAT project has received funding from the European Union's Horizon 2020 research and innovation programme under GA No. 761072

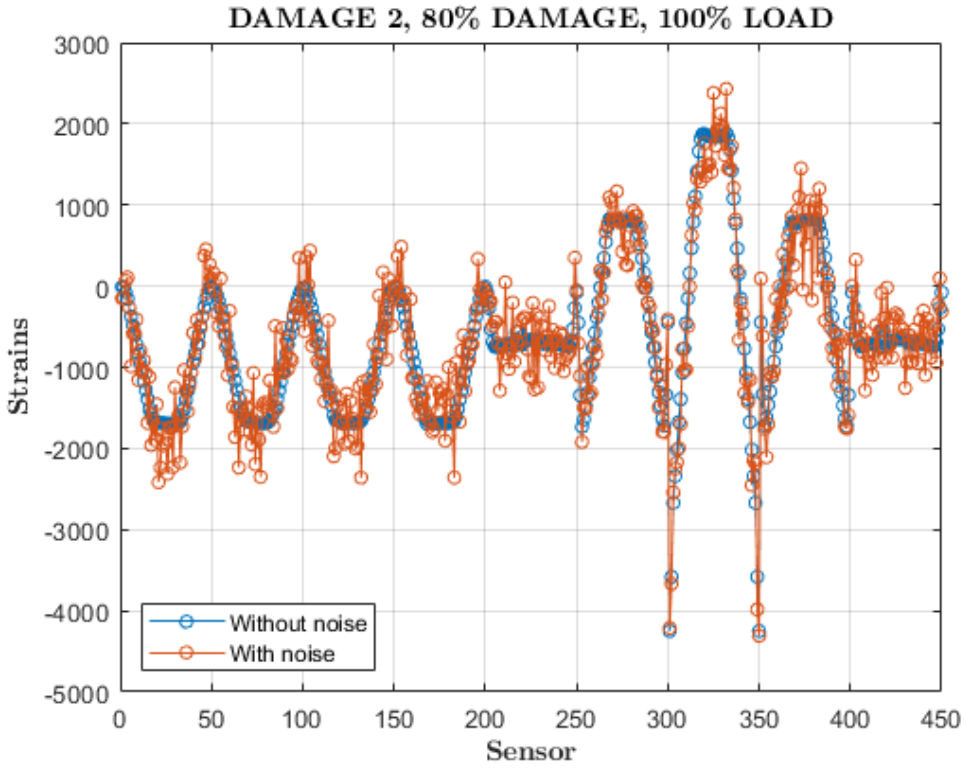


(a) Noise comparison at Damage 1

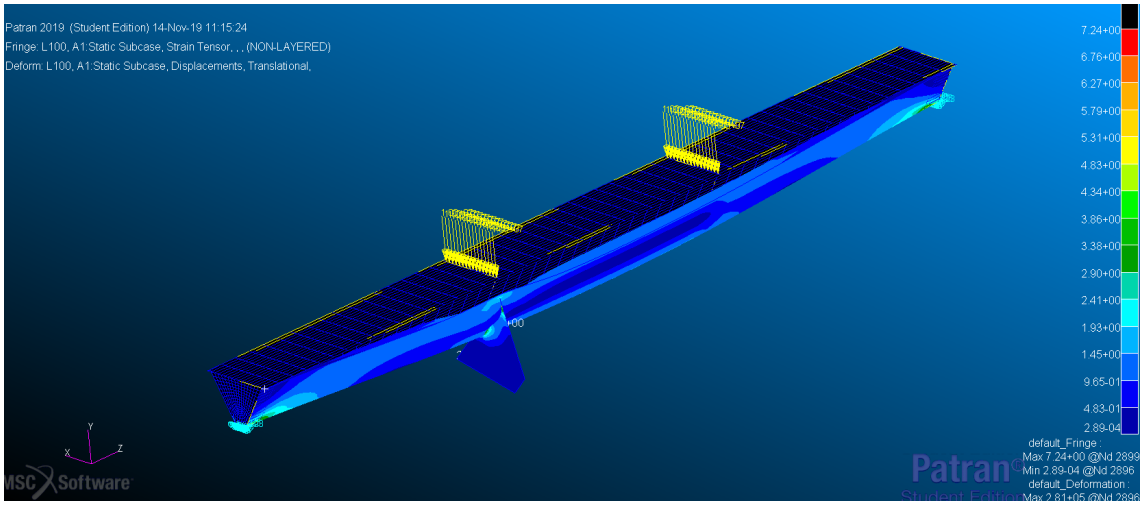


(b) Patran deformation plot

Figure 2.5.8: Damage 1 noise comparison and 3D plot



(a) Noise comparison at Damage 2



(b) Patran deformation plot

Figure 2.5.9: Damage 2 noise comparison and 3D plot

With all this information we can proceed to the algorithm definition and performance.

Damage detection algorithms

3.1 Machine Learning: LSTM network

Machine learning is a method of data analysis that automates analytical model building. It is a branch of artificial intelligence based on the idea that systems can learn from data, identify patterns and make decisions with minimal human intervention.

In this case, the structural diagnosis is carried out by an Recurrent Neural Network (RNN), proposed to solve a multiclass classification problem (Damage 1, 2 and Undamaged) with sequence data.

As with humans, RNNs don't start their thinking from scratch every second, they don't throw all the knowledge away and start thinking from scratch again. For example, if you want to classify what kind of event is happening at every scene in a movie, the most reasonable idea is to use the previous scenes in order to take a decision for the present one. This cannot be done by traditional neural network, whereas RNNs can do it. This is useful for our purpose because we will be able to classify a damage using the other damages information.

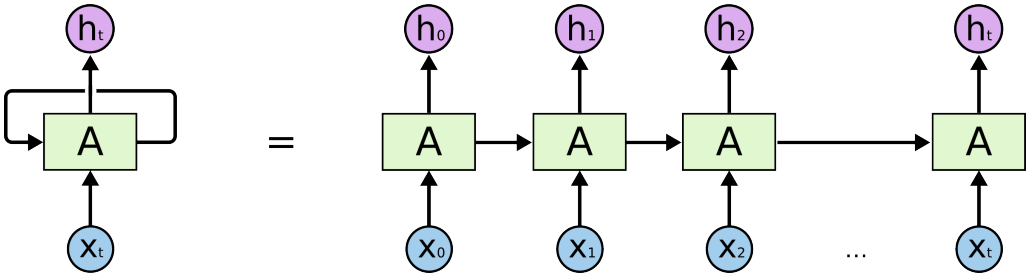


Figure 3.1.1: Recurrent Neural Network example

In the above diagram, Figure 3.1.1, shows that a recurrent neural network can be thought of as multiple copies of the same network, each passing a message, C_i, to a successor, A, which has an input x_i, and provides an output h_i.

One of the appeals of RNNs is the idea that they might be able to connect previous information to the present task. Sometimes, we only need to look at recent information to perform the present task, on the contrary there are also cases where we need more context. Unfortunately, as that gap grows, RNNs become unable to learn to connect the information.

Long Short Term Memory Networks (LSTMs) are a special kind of RNN, capable of learning long-term dependencies. They were introduced by Hochreiter & Schmidhuber (1997), and were refined and popularized by many people in following work. LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods of time is practically their default behavior. For this reason, these are the most suitable network for a long input array, like our strain field array.

Actually, we have used a biLSTM network, which has both backward and forward information propagation.

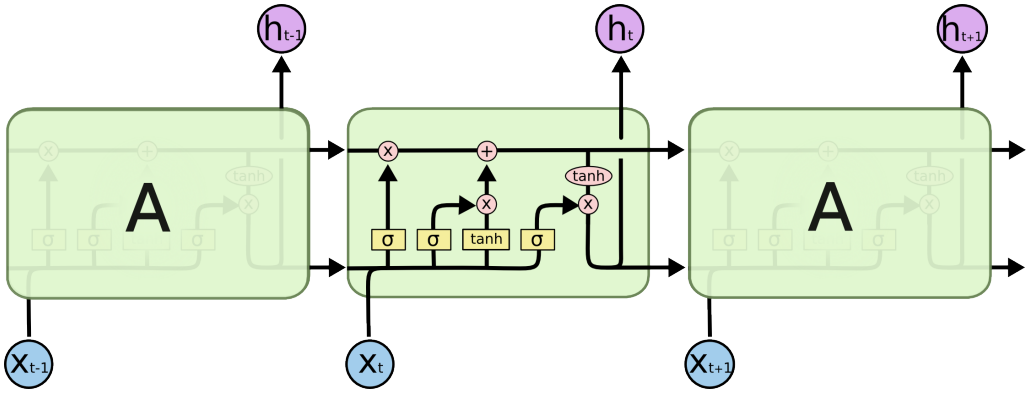


Figure 3.1.2: LSTM network example

3.1.1 Network definition

To define the layers of the network, we have to know what type of input and output we have and what type of structure admits the used LSTM layer.

The main idea of the structure state classification is that a strain field extracted from a structure (with an unknown load and damage state) is introduced to the net. The network's output will be the type of damage present in the structure, and this alone classifies the state.

- This means that an input layer is needed and this is where we define the size of the input array. In this study the size is 1, so it is only one row of information. On the contrary, if we had used both strain and stress fields, the input size would have been 2, a two row array with as many columns as elements the sensors have.

- After the input layer, we have the biLSTM layer. At this point the number of hidden blocks are defined (numHiddenUnits), and we have selected 100 of them.

- Next, a Fully Connected Layer is introduced to multiply the input for some weights belonging to each structure state, known as classes from now on.

- Then, a Softmax layer is introduced to interpret the previous data as a probability.

- Finally, a Classification Layer outputs the class as the one which has the highest probability score.

```
layers = [sequenceInputLayer(inputSize)
    bilstmLayer(numHiddenUnits,'OutputMode','last')
    fullyConnectedLayer(numClasses)
    softmaxLayer
    classificationLayer];
```

 The DACOMAT project has received funding from the European Union's Horizon 2020 research and innovation programme under GA No. 761072

```
layers =

  5x1 Layer array with layers:

     1   ''   Sequence Input          Sequence input with 1 dimensions
     2   ''   BiLSTM                 BiLSTM with 100 hidden units
     3   ''   Fully Connected        3 fully connected layer
     4   ''   Softmax                softmax
     5   ''   Classification Output   crossentropyex
```

Figure 3.1.3: Net layers

3.1.2 Training options setup

All the Neural Networks need to be trained with a classified data set to optimise the internal parameters in order to have a good performance with the desired problem. In this study, we have labelled the structure states into 3 different classes, which are: D1, D2 and Ud, and each strain array is associated with a label, Figure 3.1.4

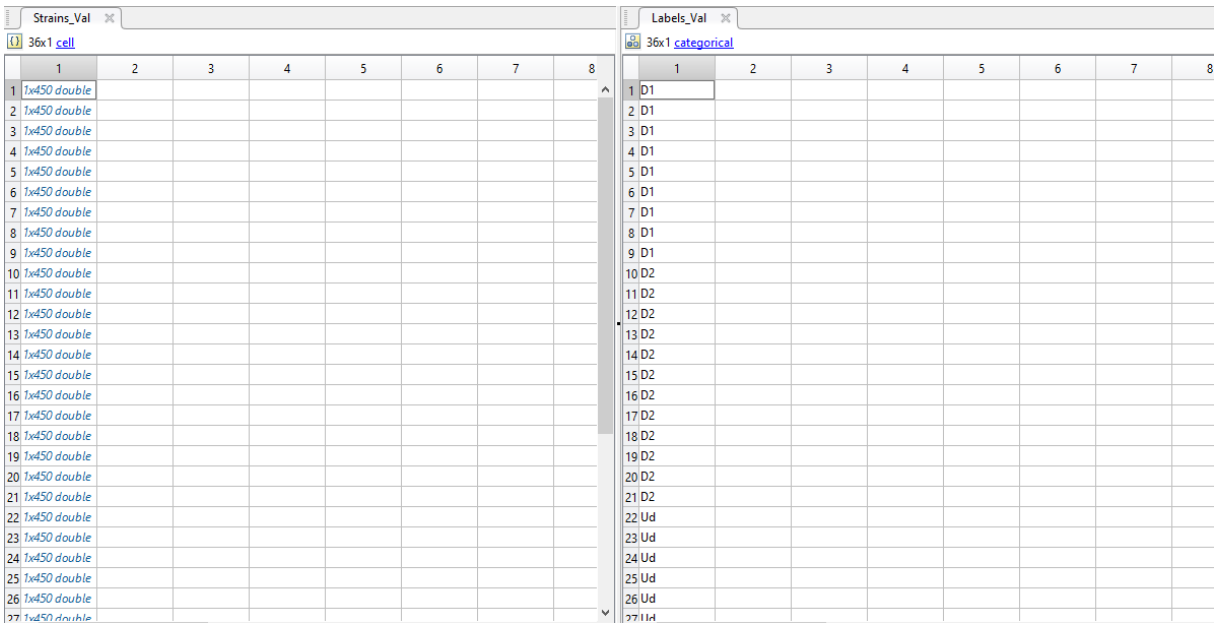


Figure 3.1.4: Validation data set

The full data set, which with all the states and the scaled loads rises up to a 240 strain field array, is divided between Training and Validation in a random division in 85% (204) and 15% (36) of the total respectively.

```
options = trainingOptions('adam', ...
        'ExecutionEnvironment','gpu', ...
        'GradientThreshold',1, ...
        'MaxEpochs',maxEpochs, ...
        'MiniBatchSize',miniBatchSize, ...
        'SequenceLength','longest', ...
        'Shuffle','never', ...
        'Verbose',0, ...
        'Plots','training-progress');
```

```
options =

  TrainingOptionsADAM with properties:

           GradientDecayFactor: 0.9000
       SquaredGradientDecayFactor: 0.9990
                         Epsilon: 1.0000e-08
                InitialLearnRate: 1.0000e-03
       LearnRateScheduleSettings: [1×1 struct]
                L2Regularization: 1.0000e-04
          GradientThresholdMethod: 'l2norm'
                GradientThreshold: 1
                        MaxEpochs: 500
                    MiniBatchSize: 30
                          Verbose: 0
                  VerboseFrequency: 50
                    ValidationData: []
                ValidationFrequency: 50
                 ValidationPatience: Inf
                          Shuffle: 'never'
                   CheckpointPath: ''
             ExecutionEnvironment: 'gpu'
                      WorkerLoad: []
                       OutputFcn: []
                           Plots: 'training-progress'
                  SequenceLength: 'longest'
             SequencePaddingValue: 0
          SequencePaddingDirection: 'right'
              DispatchInBackground: 0
           ResetInputNormalization: 1
```

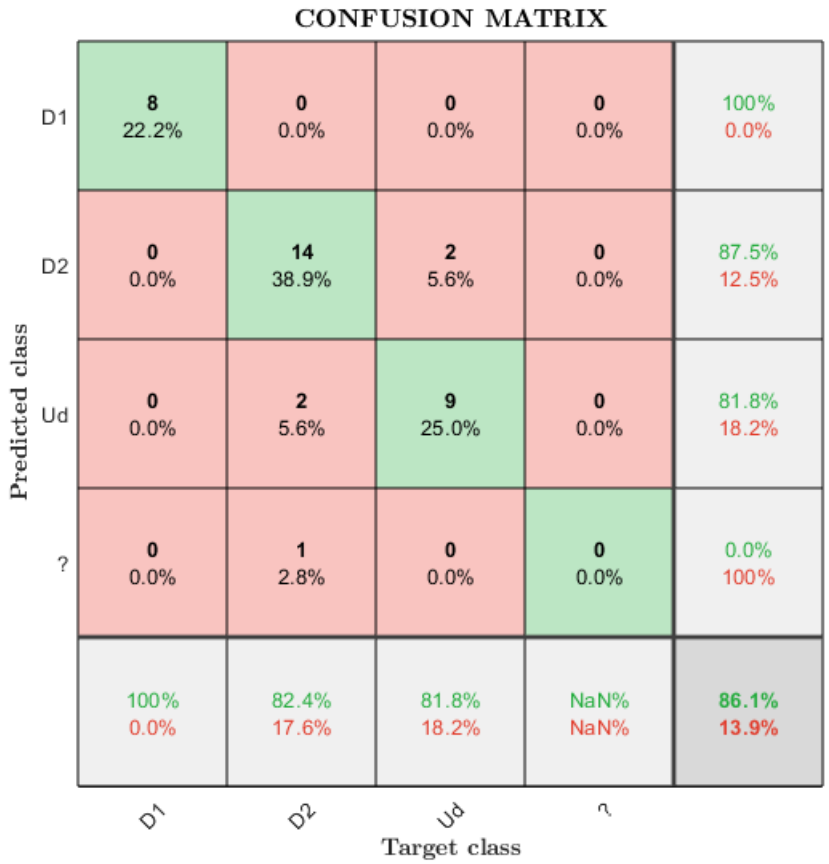
Figure 3.1.5: Training options

The Batch Size option is very useful because, at the training time, the Training data set is divided into smallest groups and the net performance increases significantly.

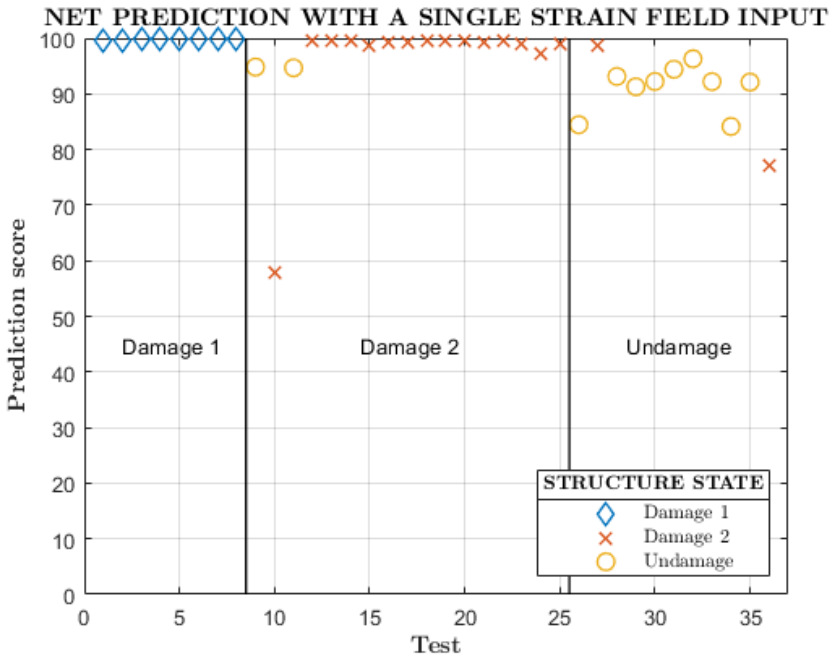
 The DACOMAT project has received funding from the European Union's Horizon 2020 research and innovation programme under GA No. 761072

3.1.3 Net performance

Once the net is trained, it is ready for the validation process. This consist of introducing the strain arrays inside the pretrained net and comparing the output label with the input one. In this way, we can have a general idea of the whole net performance, because these inputs were separated from those used to train the network. In other words, that was the first time when the net classifies this structure states.

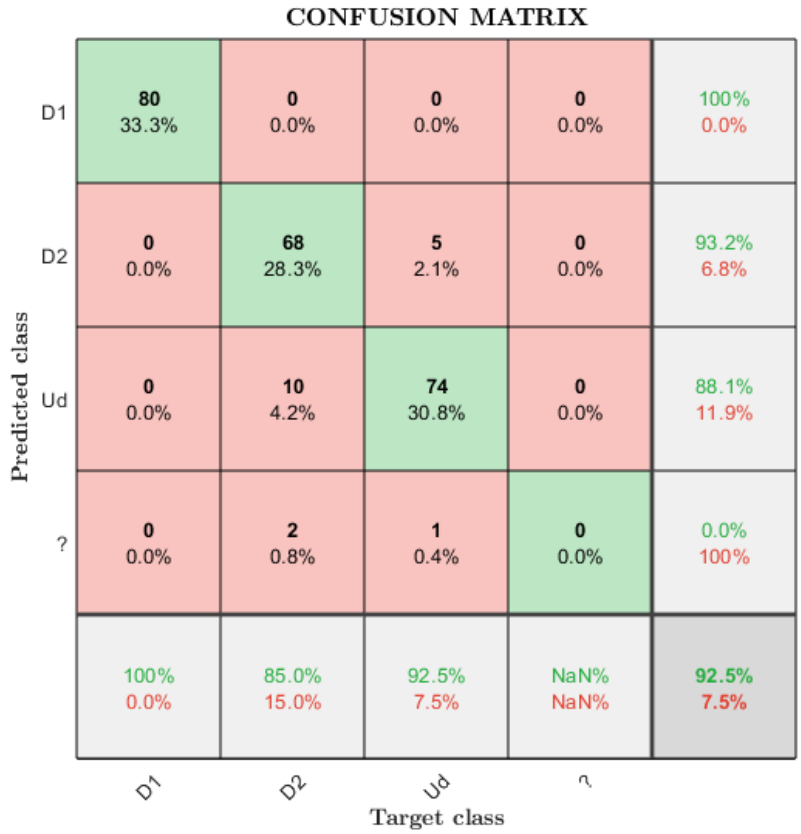


(a) Validation confusion matrix



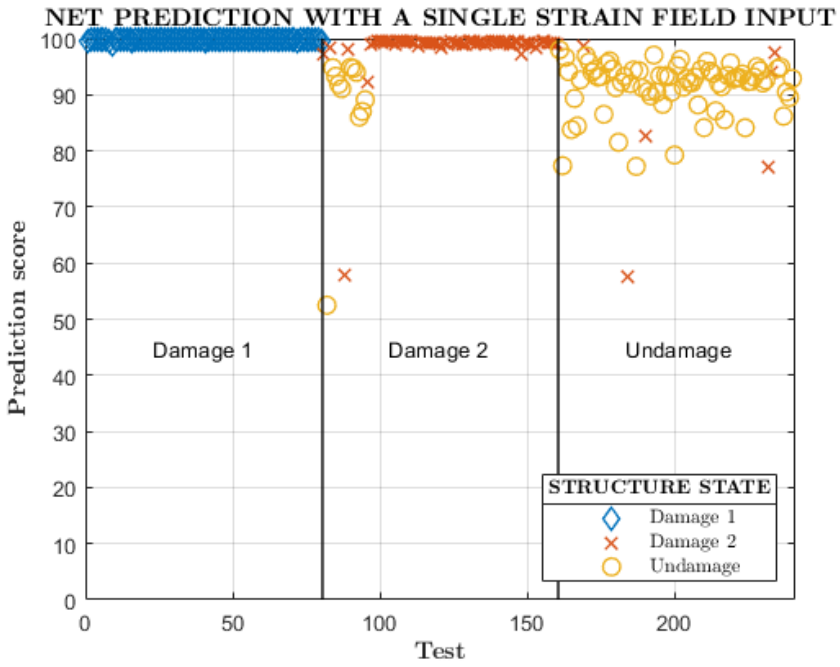
(b) Prediction scores

Figure 3.1.6: Net performance with the validation data



(a) Total data confusion matrix

 The DACOMAT project has received funding from the European Union's Horizon 2020 research and innovation programme under GA No. 761072



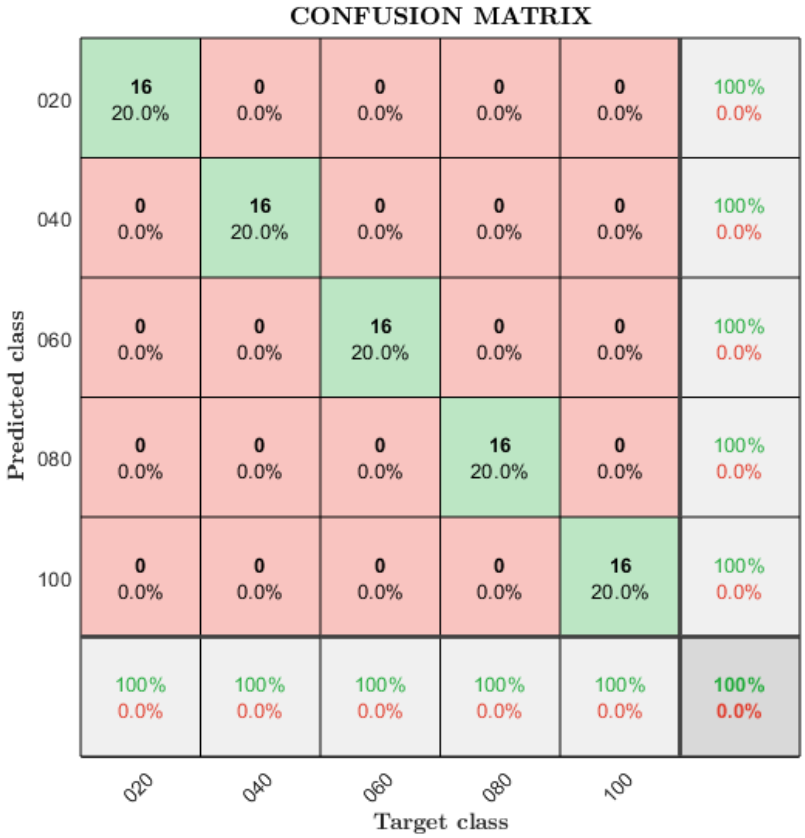
(b) Prediction scores

Figure 3.1.7: Net performance with the all the data

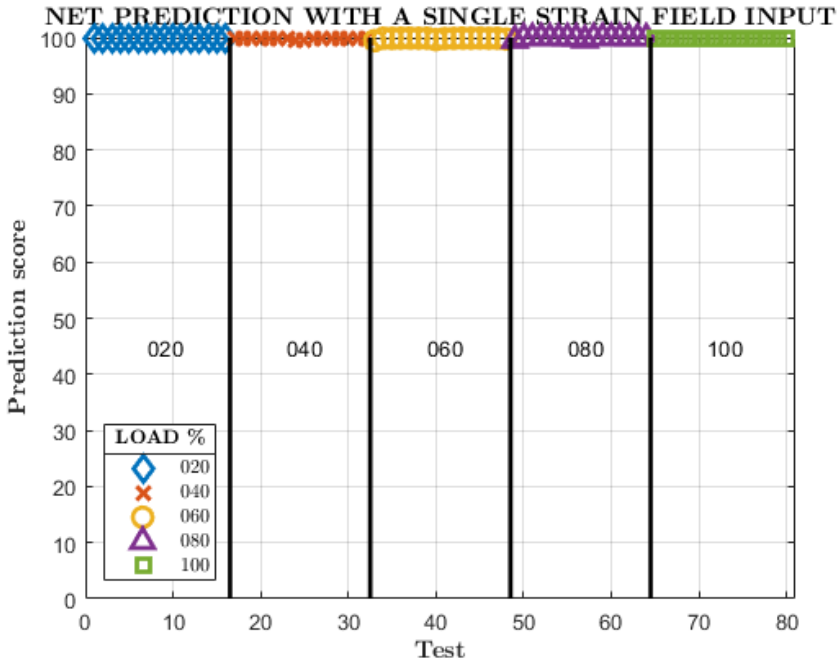
In Figures 3.1.6 and 3.1.7 the net performance can be compared between the Validation and Total data input. The net accuracy is obviously better in the second one because it has been trained with a huge amount of cases introduced to test it. It is important to emphasise that when a predicted score is less than 60% it has been classified as indecisive, "?"

In the same way, a network orientated to detect the debond size has been trained for the two damaged states. However, the results in Figures 3.1.8 and 3.1.9 are obtained with all the load cases instead of doing it with the validation data. The reason for having done this is because with only 16 strain fields at each load case, 80 in total, 15% in not a representative number of tests in order to have a representative view of the net performance.

The DACOMAT project has received funding from the European Union's Horizon 2020 research and innovation programme under GA No. 761072

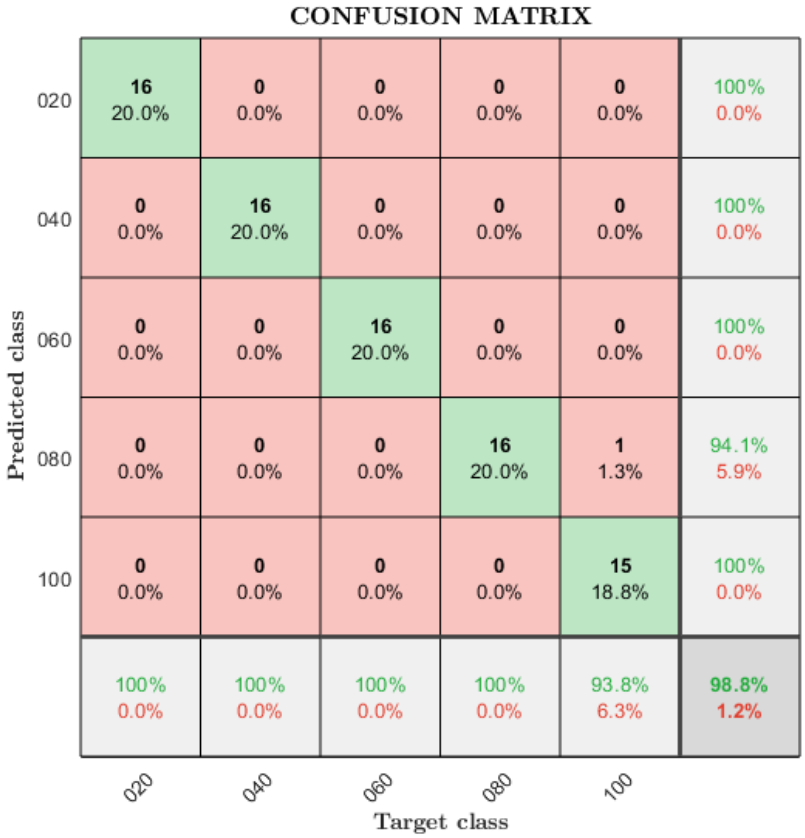


(a) Top surface debond confusion matrix

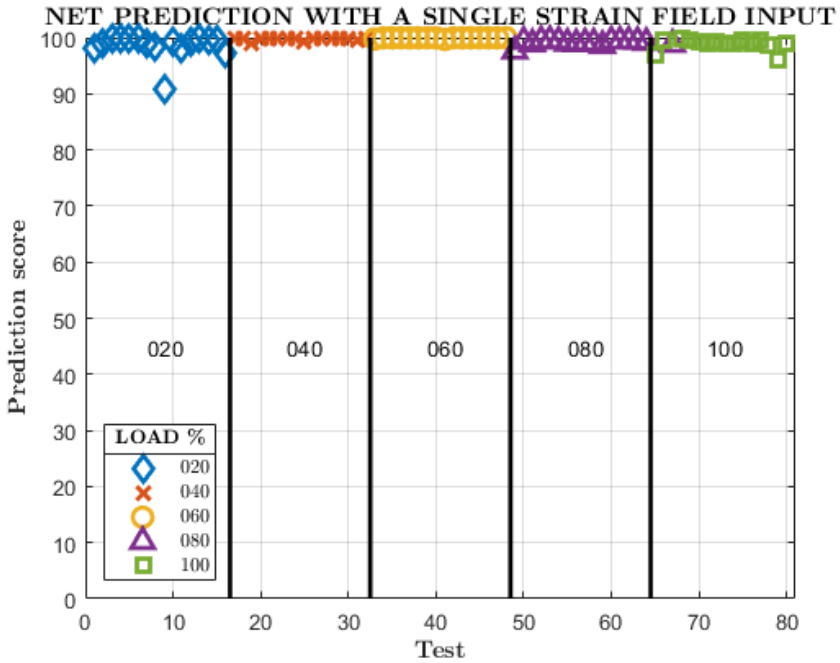


(b) Prediction scores

Figure 3.1.8: Classification of increasing Damage 1



(a) Second rib debond confusion matrix



(b) Prediction scores

Figure 3.1.9: Classification of increasing Damage 2

3.2 Principal Component Analysis (PCA)

PCA is a widely use non-parametric method of extracting information from large data sets. It is a classical multivariate analysis procedure to reduce a complex data set to a lower dimension and reveal some hidden structure/pattern. The original data are reexpressed in a new orthogonal basis where the data are arranged along directions of maximal variance and minimal redundancy, called principal components. The damage indicator for this technique is the Q-index and the detection capability has been proven at different damage states using FBG strain with the fully loaded structure. Although Q-index is a powerful damage indicator, capable of detecting damage increments, it cannot be used to determine damage location or accurate damage evolution. However, due to its easy implementation and fast processing, it is a very widespread SHM analysis method. A detailed discussion was formerly presented at APWSHM 2018 (DOI: 10.1177/1475921710388972) and will not be repeated here.

PCA is a mathematical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components. This transformation is defined in such a way that the first principal component has the largest possible variance (that is, accounts for as much of the variability in the data as possible). Usually, the number of principal components can be much smaller than the number of original variables. Each succeeding component in turn has the highest variance possible under the constraint that it be orthogonal to (that is, uncorrelated with) the preceding components. The PCA algorithm is composed of the following mathematics operations (Matlab tools are available for it):

1. Organize the data set as n x m matrix, where n is the number of experiments and m is the number of measured variables: X

2. Normalize the data to have a zero mean and unity variance

3. Calculate the eigenvectors-eigenvalues of the covariance matrix: $C = XX^T$

4. Keep only the first eigenvectors as the principal components

5. Project any new collected data into the former baseline

6. Identify if new data follow global trends (Damage Index)

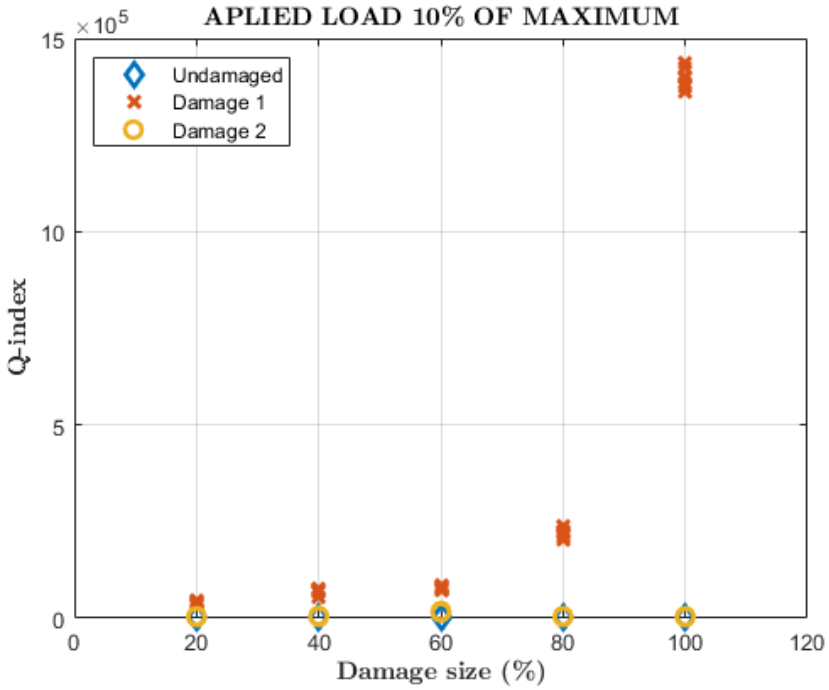
There are statistical tools that, used along with PCA, allow the detection of anomalous behavior in systems. The two most common tools are the Q index (or index SPE Squared Prediction Error) and the T-index. The index Q indicates how well each sample fits the PCA model. It is a measure of the difference between a sample and its projection in the main components retained by the PCA model.

3.2.1 PCA performance

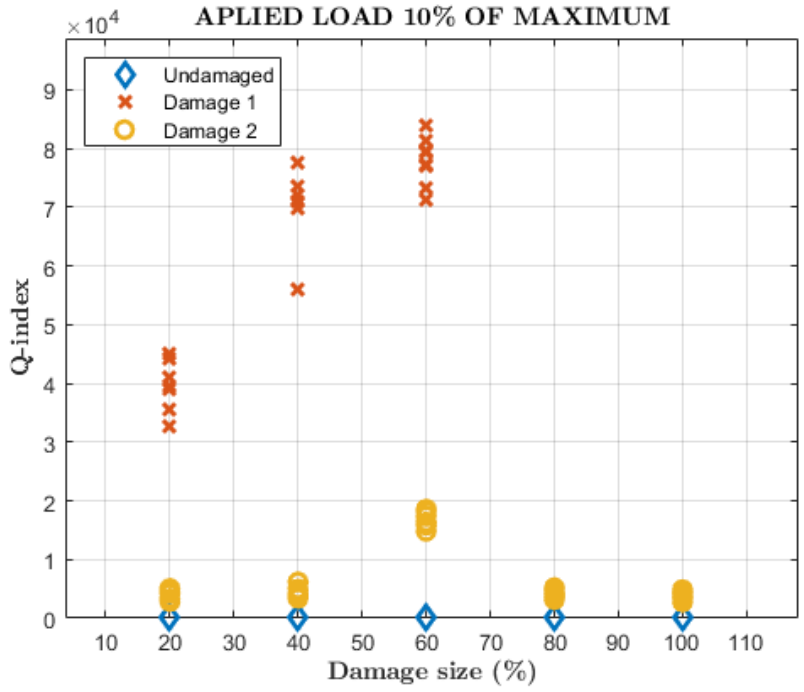
In the next figures (3.2.1) the Q index evolution is shown with an increased debonding of the three structure states.

The first figure shows that the Damage 1 has a higher Q index than the other two states and it follows the expected increasing tendency. The second figure is a detail view of the first one. Here a big dispersion of the Damage 2 Q-index can be seen, and it is difficult to differentiate it from the undamaged state.

 The DACOMAT project has received funding from the European Union's Horizon 2020 research and innovation programme under GA No. 761072



(a) Ud, D1 and D2 Q index



(b) Q index detail

Figure 3.2.1: Q index projected to Undamaged subspace

Conclusions

The benefits envisioned by the SHM approach for civil structures inspections without disassembly include minimising costs, improving reliability, and maximising availability of aircraft. Based on previous results, it can be concluded:

- FOS are excellent strain sensors, but not 'damage sensors'.
- FOS are a good option for SHM on civil structures.
- PCA provides an index to calculate the damage occurrence. However, this technique can not calculate the damage location or assess damage size.
- RNN requires a previous training and is more difficult to use and compute.
- RNN could provide not only the damage assessment, but could also provide a detailed localisation of the damage area that can be presented as a damage map.
- LSTM network has the capability to detect the presence of a damage and quantify this damage (%).
- The capability of learning load dependencies at different damage states has been demonstrated by LSTM network.
- LSTM network can perform an accurate structure state classification.
- The LSTM network cannot make a clear distinction between Damage 2 and the Undamaged state due to the fact that the deformations are practically identical, Figure 4.0.1.

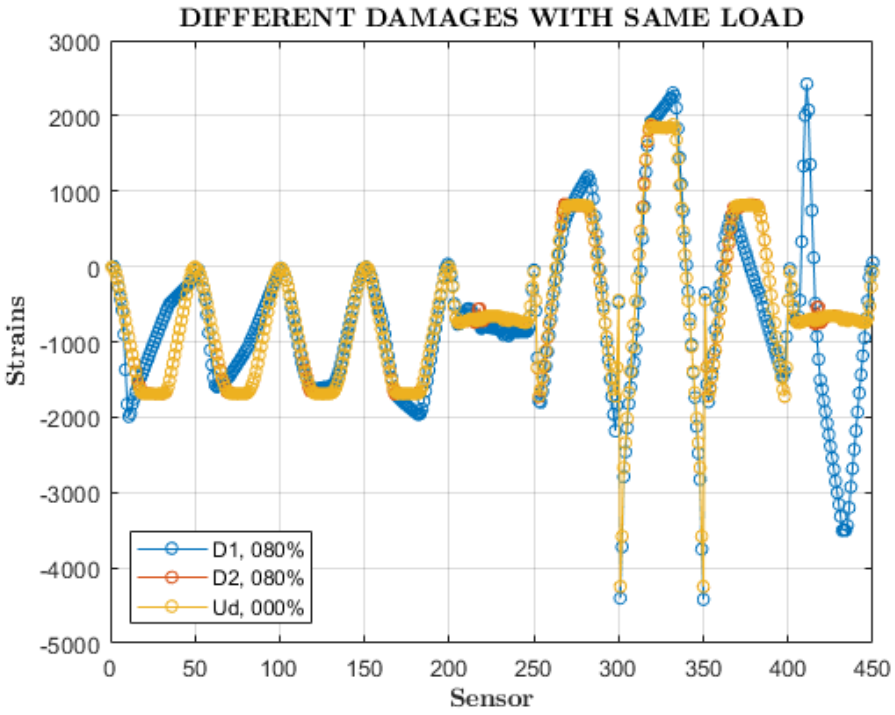


Figure 4.0.1: Comparison among different structure strain fields

- The similarity of the deformation fields is due to the fact that the ribs are an element that stiffens to torsion, while the beam is working in flexion. This makes its debonding not as relevant as that of the top cover.
- The similarity between the stain field caused by a same load at different debond lengths (Figure 2.5.5c) makes the trained network unable to differentiate between some debond lengths.
- The RNN validation with a FEM gives a first approach to the possibilities of the algorithm, the next step should be checking the algorithm performance on a real structure.
- A structure state classification algorithm was implemented but, following the same steps, a load state detection can be done.

The DACOMAT project has received funding from the European Union's Horizon 2020 research and innovation programme under GA No. 761072