# How fraud detection technologies can help to detect damages in aircraft structures

Arnaud Cugniere[1], Olaf Tusch[2], and Andreas Mösenbacher[2]

[1]IABG mbh
[2]IABG mbH

July 16, 2020

**Abstract**

A software architecture based on Machine Learning (ML) and Finite Element Method (FEM) and aimed at improving the detection of damages in aircraft structure subjected to complex variable loadings is presented here. Firstly, the software relies on statistical tools used among others in fraud detection (One-Class Support Vector Machine, Local Outlier Factors, Isolation Forest, DBSCAN) to identify anomalies in a vast amount of data recorded over time by multiple strain gauges located on the structure of the aircraft. Once an anomaly is detected at a given time and for a specific set of strain gauges, it can be classified as insignificant or critical by the user. If the anomaly is critical, the data of the associated strain gauges can be used as input data for a FEM optimization. This static optimization allows to visually assess the position and geometry of possible cracks in the structure.

**KEYWORDS**

Machine learning, finite element modelling, crack identification, aircraft structure, structure optimization, data mining, big data, digitalization.

## INTRODUCTION

During their lifespan, aircraft structures are subject to significant loadings that generate, on the long-term, fatigue damage .

In order to obtain a certification that guarantees the safety of the aircraft, a comprehensive set of tests (e.g. fatigue testing) have to be performed. Whole ranges of flight conditions are repeatedly simulated in testing facilities, primarily to validate the faithfulness of the virtual models representing the structure but also to demonstrate the tolerance of the real structure with regard to damage.

To ascertain the presence or absence of cracks in the structure, a limited number of strain gauges are disposed at key positions. These strain gauges provide information about the current state of the structure after a number of load cycles.

Cracks can sometimes appear earlier than expected and still be undetected until the first scheduled inspection. However, a substantial analysis of the data recorded by the strain gauges, using computational methods, could prove useful to identify hidden cracks much earlier than by visually inspecting the structure or by manually searching for noticeable variations in the recorded data.

One of those computational methods is called anomaly detection, an approach widely used nowadays in fraud detection.

Presented here is a software architecture based on ML and FEM that tries to encapsulate this approach.

An early detection of these cracks based on this modus operandi means a better understanding of the underlying mechanisms of cracks propagation.

## TEST CONFIGURATION

To be able to proof the validity of the concept, data from a test campaign conducted over many years were used. This test campaign was conducted in the scope of a certification process of an aircraft structure.

The aircraft structure is located in a hangar where the test campaign is carried out.

The test rig comprises several hydraulic actuators in contact with the aircraft structure. These hydraulic actuators apply specific forces on the structure for a given load case. Each load case represents a specific phase (take-off, landing . . . ). Different load cases are combined in a so-called loading program, which represents the complete life of the aircraft structure. A test campaign can be seen as a faithful representation of what an aircraft would experience during its lifetime. Therefore, it combines different flight conditions in a realistic way, which roughly means simulating many flights in normal conditions and few in strong, tough and extreme conditions.

In order to keep track of the number of flights simulated during the test campaign so far, a flight counter was implemented.

Several strain gauges were disposed at key positions on the surface of the structure and were used to record the intensity of the strains for each simulated flight.

## ANOMALY DETECTION

As mentioned before, up to now, in the certification phase of the aircraft, crack detection has depended essentially upon visual inspection, which has made the process difficult, for it has relied solely on the ability of the technician to spot macroscopic cracks on an, in comparison, extremely large structure.

A data-driven approach, on the contrary, provides an automatic way to do that, with potentially the prospect of detecting cracks earlier and more reliably than with the traditional approach.

Utilizing the data provided by the strain gauges as a reference to detect cracks implies several assumptions:

- Cracks that occurred prior to the test campaign will not be detected.
- The occurrence of a crack near strain gauges has a significant and measurable effect on their recorded data.
- As a crack propagates, changes in the recorded data will be detected over time.

The main idea of the approach is the following: when a crack occurs in the aircraft structure within the sensitivity range of a strain gauge, a corresponding anomaly in the recorded data will be detected.

In a classical sense, an anomaly refers to something that is different from what is usual or expected. Anomaly detection is the process of identifying unexpected items or events in datasets, which differs from the norm. Anomaly detection is commonly used in fraud detection or network security . In fraud detection for banking for instance, excessive money withdrawals in a short period of time might reflect something unusual and might be a clue to a credit card theft. Using anomaly detection, this kind of unusual behaviours can be identified automatically and rapidly.

In a statistical sense though, there are different definitions of an anomaly (also called outliers) . In figure *1* for example, where points from a dummy dataset with two features (x, y) are represented, there is a clear separation between the data, with two significant, well separated clusters (c1, c2) and a third sparse, smaller, isolated cluster (c3 + c4) that can be seen as a global anomaly. On the scale of the third cluster though, point c4 can be regarded as a local anomaly with regard to cluster c3, which can be seen in this case as normal data.

1. Global vs. Local anomalies

Here is important to distinguish between what is considered to be "normal" data (data that reflect the correct functioning of a system and shouldn't be considered anomalies) and anomalies (data that reflect a malfunctioning of a system). In some cases, the amount of anomalies can outnumber the amount of normal data.

For this reason, depending on whether global or local anomalies are to be found, different anomaly detection algorithms will be preferred.

The type of training data available is also a critical factor in choosing the right algorithm. There are three modes of anomaly detections (supervised, semi-supervised & unsupervised learning) depending on whether:

- the training data contains both normal data and anomalies (supervised learning)
- the training data contains only normal data (semi-supervised)
- no training data is available at all (unsupervised).

Figure *2* sums up the different algorithms available for each mode:

2. Anomaly detection modes

For this problem, semi-supervised and unsupervised approaches were chosen:

*For the semi-supervised learning approach* :

*Local Outlier Factor (LOF)* . LOFs are k-Nearest-Neighbours-based algorithms, which means that these algorithms assume that outliers lie in sparse neighbourhoods and are far away from their nearest-neighbours . In simple terms, what LOF does when probing a data point is simply looking at the density of the distribution around this point and comparing it with the density of the distribution around each neighbouring point. If the density around the probed point is similar to the density measured at the neighbouring points, it means that the probed point belongs to a cluster formed by those neighbouring points. If the densities are different, it means that this point is probably an outlier. Figure *3* shows a graphical representation of this concept:

3. Main concept of Local Outlier Factor

*Isolation Forest* . Like LOF, Isolation Forest aims to detect outliers in a feature-space. It is based on the following steps: select a feature and randomly split this feature at a value located between the minimum and maximum data points. In this way, all data points are separated into two groups. Each group is then randomly split again in the same manner. This process is repeated iteratively until all data points are isolated (no more splitting possible). At last, these successive splitting steps form a structure comparable to a tree, from the root node, which corresponds to the initial splitting, to the leaf, which corresponds an isolated data point, each leaf representing a different data point. The higher the branch to the leaf, the harder it is to isolate the data point (that means the data point probably corresponds to normal data). The shorter the branch to the leaf, the easier it is to isolate the data point (that means the data point probably corresponds to an outlier). Figure *4* shows a graphical representation of this concept:

4. Main concept of Isolation Forest

*One-Class Support Vector Machine (One-Class SVM)* . One-class SVMs attempt to learn a decision boundary that achieves the maximum separation between the data points and the origin . One-Class SVMs are derived from classical SVMs, which are powerful algorithms that attempt to separate points of different classes in a feature space by fitting a decision boundary between the different classes. In a 2D space with two classes of data points for instance, the SVM algorithm tries to find the optimal position of a 2D-curve that separates the two classes with the maximal possible margin between them. Figure *5* shows a graphical representation of this concept:

5. Main concept of Support Vector Machine

In case the classes cannot be separated by a simple straight line, SVM can transform the feature space into a higher-dimensional space (2D to 3D for instance) by using a so-called Kernel function . Once in the 3D

space, it can split the classes with a 3D plane and project this plane back into a 2D space and get a 2D-circle that separates the two classes with the maximal possible margin. Figure *6* shows a graphical representation of this concept:

6. Main concept of Support Vector Machine and Kernel principle

*For the unsupervised learning approach* :

*DBSCAN* . DBSCAN is a clustering algorithm that clusters the data and measures the distance from each instance to its nearest cluster centre. In this case, the clustering approach is not meant as a way of identifying outliers but rather as a way of understanding how the data points cluster with each other (in other terms, how the data is arranged)

Such configuration allows to compare the efficiency of global versus local anomaly detection algorithms. Using a dummy dataset similar to the one represented in Figure 1, the three semi-supervised approaches are compared in figure *7* :

7. Comparison between Anomaly Detection Algorithms for a dummy dataset

Similarly, the unsupervised approach can be tested on the same dataset in figure *8* :

8. Clustering of a dummy dataset using DBSCAN

## DIMENSIONALITY REDUCTION

The data recorded by the strain gauges over the test campaign shows that the strain gauges experience strong discrepancies over time. However, these discrepancies are, first and foremost, periodic. In other terms, an amplitude that goes up will decrease at some point. Therefore, it is not possible to use a maximum threshold beyond which a strain amplitude can be seen as critical and be associated with a crack. Figure 9 shows the amplitude of one strain gauge over several months, for one particular load case:

9. Amplitude from a strain gauge over several months for a specific load case

Figure 10 helps to visualize how the measured strains vary over time despite no changes in the loading conditions:

- The upper diagram in figure 10 represents the amplitude measured by one particular strain gauge for several load cases. The x-axis represents 700 different load cases that were simulated in the test rig. The y-axis represents the strain amplitude. The colour scale represents the flight count (˜time) (the blueish lines correspond to the beginning of the test campaign, the yellowish lines correspond to the middle of the test campaign and the reddish lines correspond to the end of the test campaign). There are obviously fluctuations over time because the reddish, yellowish and blueish lines are relatively well separated.
- The lower diagram in figure 10 represents the amplitude of the pressure that is applied inside the cabin. The x-axis represents the 700 load cases, the y-axis represents the pressure amplitude. Like in the upper diagram, the colour scale represents the flight count. There is obviously no fluctuation in this diagram, because the curves are overlapping.
- By comparing those two diagrams, it becomes clear that the strain fluctuates over time despite no fluctuation in the loading:

10. Amplitude recorded by a strain gauge and a pressure gauge inside the cabin over a period of several months

Although the fluctuations of the strain amplitude over time are obvious in the upper diagram, they strongly vary depending upon which load case is considered. For instance, the load cases where high cabin pressure was applied show high fluctuations of the strain amplitude over time (the reddish, yellowish and blueish lines in the upper diagram are well separated). On the contrary, the load cases where low cabin pressure was

applied show only slight fluctuations of the strain amplitude over time (the reddish, yellowish and blueish lines in the upper diagram are almost overlapping).

An assumption was made that no information about the loading conditions would be made available. This means, the anomaly detection algorithm has to work exclusively with the strain amplitudes of the strain gauges and not with any measures of cabin pressure or hydraulic cylinder's forces.

An approach based on Fast-Fourier-Transform (FFT) was used because it allows to capture the changes that affect the strain gauges over time, independently of the load cases. Figure 11 is a representation of the approach:

11. Pipeline "anomaly detection" for a specific strain gauge

The main idea here is to analyse the raw signal recorded by a strain gauge at each flight (each time iteration). In the raw signal, the x-axis doesn't represent time but rather the simulated load cases (like in the upper diagram in figure 10). This signal is analysed using a FFT-based spectral analysis. The results of the spectral analysis is a spectral diagram where the x-axis doesn't represent frequency per se but rather the terms of the FFT. Throughout the test campaign, as cracks appear in the structure, the results of the spectral analysis will change. In other terms, fluctuations recorded in the raw signals over time will be seen in the different spectral diagrams over time. Moreover, in the spectral diagrams, instead of looking at all the terms of the FFT, one can look at just a few terms, hence reducing the number of features. In this project, this step was called dimensionality reduction.

The dimensionality reduction algorithm was combined with the several anomaly detection algorithms to form the "anomaly detection" pipeline.

Figure *12* shows the results of the anomaly detection pipeline for one particular strain gauge:

- Each point of the dataset represents a particular flight count (˜ a particular time stamp)
- The X-, Y- and Z-axes represent respectively the low, mid and high terms of the FFT-based spectral analysis.
- The data points build up clusters that are then analysed by the three semi-supervised anomaly detection algorithms (One-Class SVM, Isolation Forest and Local Outlier Factor). Each point is classified either as normal data or anomaly, depending on the cluster they belong to.

12. Results from anomaly detection pipeline for one particular strain gauge

A test was performed with data from 45 different strain gauges. The goal was to see whether anomalies could be detected for a period ranging from 0 to 15000 simulated flights. For the semi-supervised approaches, the training data comprise the period ranging from 0 to 1000 simulated flights. Figure *13* shows the results produced by the three anomaly detection algorithms (red: normal data; blue: anomalies; white: missing data) and the clustering algorithm (yellow: missing data; other colours: different clusters):

13. Results from anomaly detection pipeline

The different methods showed similar results: anomalies were detected for almost all strain gauges early on in the test campaign. Those anomalies could be attributed to either:

- Malfunctioning of the strain gauges
- Intentional operational actions affecting the strain gauges (disconnection, repositioning, malfunction, . . . )
- Strong thermal changes in the hangar (where the tests are conducted) that lead to elastic thermal expansion
- Cracks

## CRACK QUANTIFICATION AND LOCALIZATION WITH FEM PIPELINE

As previously mentioned, an anomaly can potentially be associated with a crack. In order to quantify and localize this potential crack, the information recorded by the corresponding strain gauges could be sent to an additional pipeline: the "FEM" pipeline

A precondition for using the "FEM" pipeline is the existence of a static FEM-model of the aircraft structure. The main idea here consists in using design optimization techniques (normally used to create "better" structures) to make the structure "worse" instead.

Broadly speaking, a design optimization relies on three blocks: the objective function (which quantity needs to be maximized or minimized?), the design variables (what can be changed in the design?) and the design constraints (which conditions need to be respected?).

- Here, the *objective function* is defined as a slight volume reduction (around 0.001% of the initial total volume) that accounts for the reduction of rigidity at the location of the crack.
- The *design variables* are the stiffness of each element. For 2D-shell elements, that is equivalent to optimizing the thickness of the elements.
- The *design constraints* are the strains recorded at the time of the anomaly. The position of each strain gauge on the real structure is known and can be associated with a corresponding element in the virtual structure. In this way, the strain distribution in the virtual model ought to be similar to the strain distribution in the damaged structure.

This FEM pipeline has not been implemented yet. To validate the approach, a proof of concept was carried out: a 2D-model of a 1.6-mm-thick probe was created, with the load case described in figure*14* :

14. 2D Model of probe with static load case

The distribution of von Mises equivalent strains along the structure was calculated for both the initial state (without crack) and a damaged state (with a 10mm-long synthetic crack), as can be seen in figure*15* :

15. Distribution of von Mises equivalent strains for undamaged and damaged states

Using the strain distribution of the damaged structure as design constraints, an optimization was performed on the undamaged structure. The thickness of every 2D-shell element of the undamaged structure was defined as design variable. The goal was to see whether the optimizer could retrieve the position of the synthetic crack. Figure *16*shows the results of the optimization, compared to the expected results:

16. Results of the topometry optimization vs. expected results

The optimizer chooses to minimize the thickness of the elements that roughly match the position of the synthetic crack (red = thickness unchanged, yellow = thickness reduced). Therefore, the approach can be seen as valid, with high potential in other applications.

## CONCLUSION

A framework that would combine these two pipelines could drastically simplify and accelerate the certification process of aircraft structures. In addition, it could be extended to a broad range of industries where data-driven Structure Health Monitoring (SHM) is essential. IABG is currently closely involved in the development of such a framework.

## REFERENCE

[1] J. Schijve, ,,The accumulation of fatigue damage in aircraft materials and structures,“ Advisory Group for Aerospace I

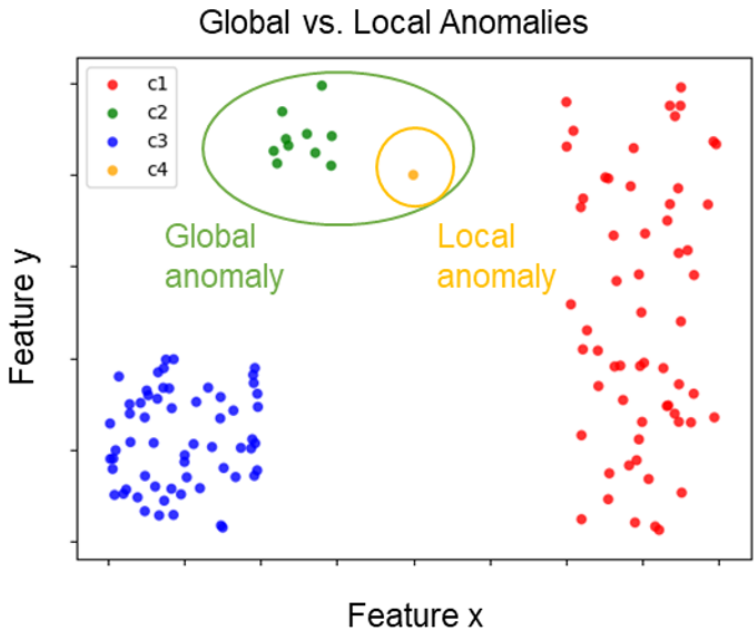
[2] M. Goldstein und U. Seiichi, ,,A comparative Evaluation of Unsupervised Anomaly Detection Algorithms for Multivari
[3] Yufeng Kou, Chang-Tien Lu, S. Sirwongwattana und Yo-Ping Huang, ,,Survey of fraud detection techniques,“ *IEEE I*
[4] P.García-Teodoroa, J.Díaz-Verdejoa, G.Maciá-Fernándeza und E.Vázquezb, ,,Anomaly-based network intrusion detecti
[5] A. Mennatallah und G. Markus, ,,Nearest-Neighbor and Clustering based Anomaly Detection Algorithms for RapidMi
[6] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, a. J. Smola und R. C. Williamson., ,,Estimating the support of a high-dime
[7] B. Schölkopf, R. C. Williamson, A. J.Smola, J. Shawe-Taylor und J. C. Platt, ,,Support vector method for novelty dete

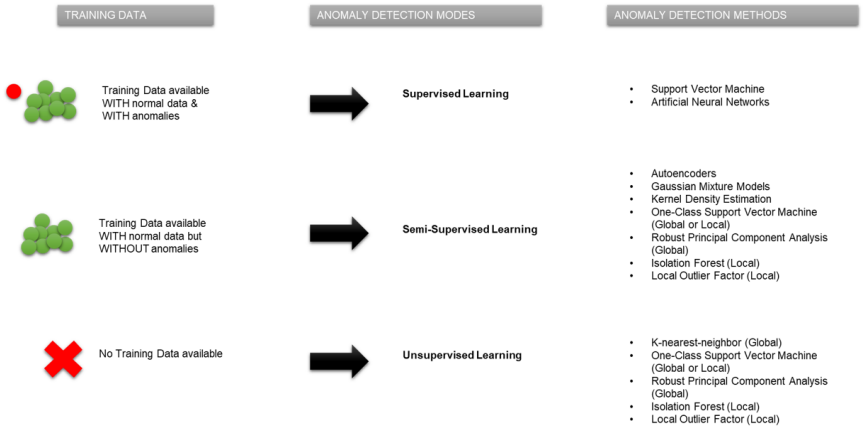
[1] J. Schijve, ,,The accumulation of fatigue damage in aircraft materials and structures,“ Advisory Group for Aerospace I

[8] G. Zhang, ,,medium.com, What is the kernel trick? Why is it important?,“ 11 November 2018. [Online]. Available: ht

[9] A. Cugniere, O. Tusch und A. Mösenbacher, ,,Early detection of damages in aircraft structures using Machine Learnin

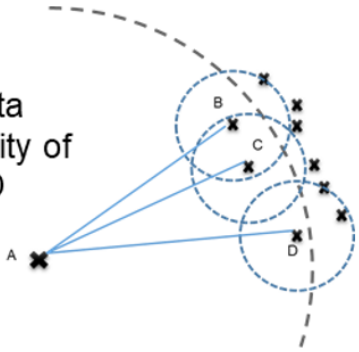
**Corresponding author:** cugniere@iabg.de



***Local Outlier Factor***

The local density around the data point A is compared to the density of its neighbouring points B, C & D

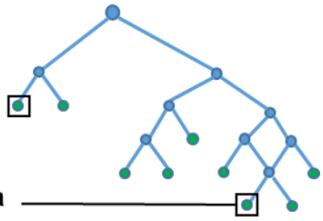



***Isolation Forest***

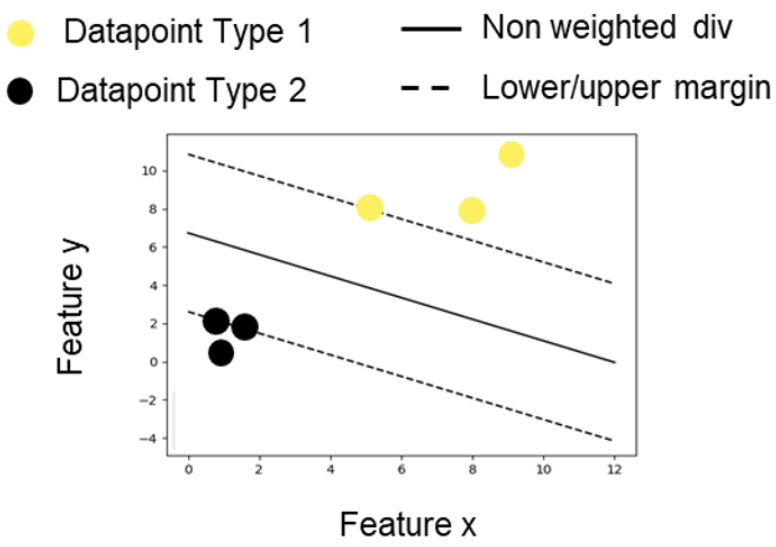
Data points that correspond to **outliers** are easier to isolate

**Depth**

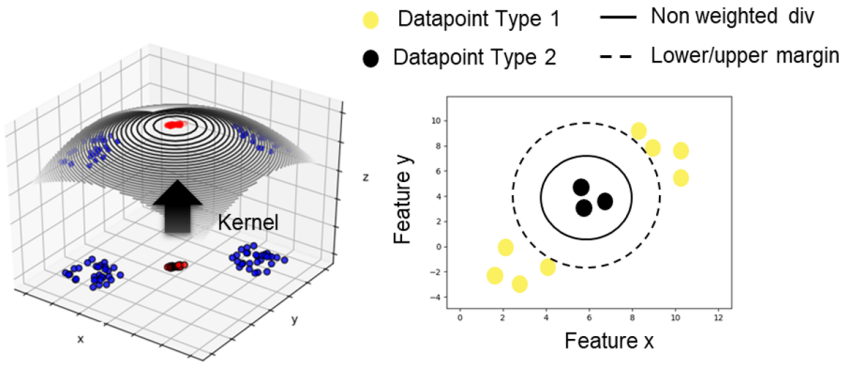
Data points that correspond to **normal data** are harder to isolate

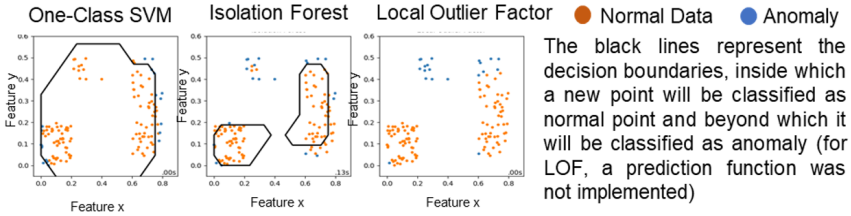


***Support Vector Machine***

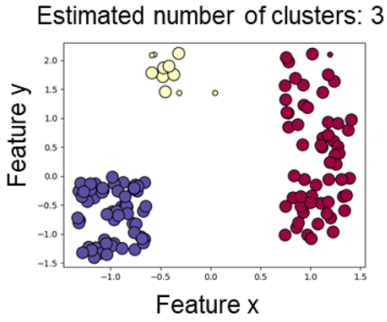


***Support Vector Machine***

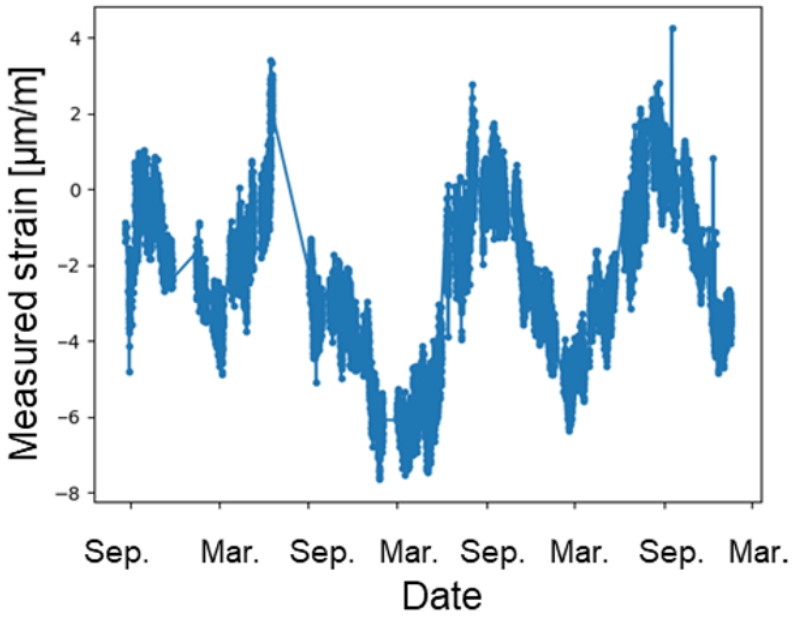


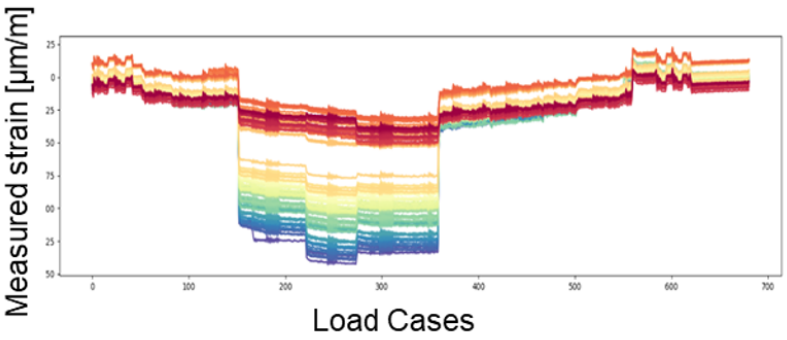



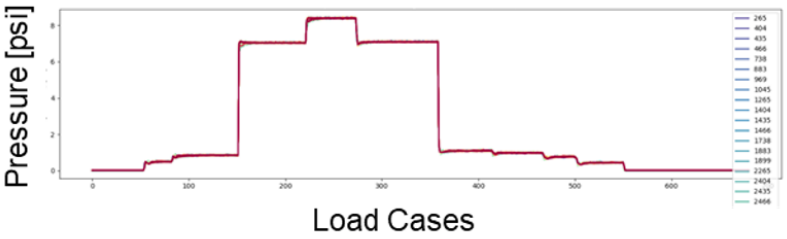

The black lines represent the decision boundaries, inside which a new point will be classified as normal point and beyond which it will be classified as anomaly (for LOF, a prediction function was not implemented)

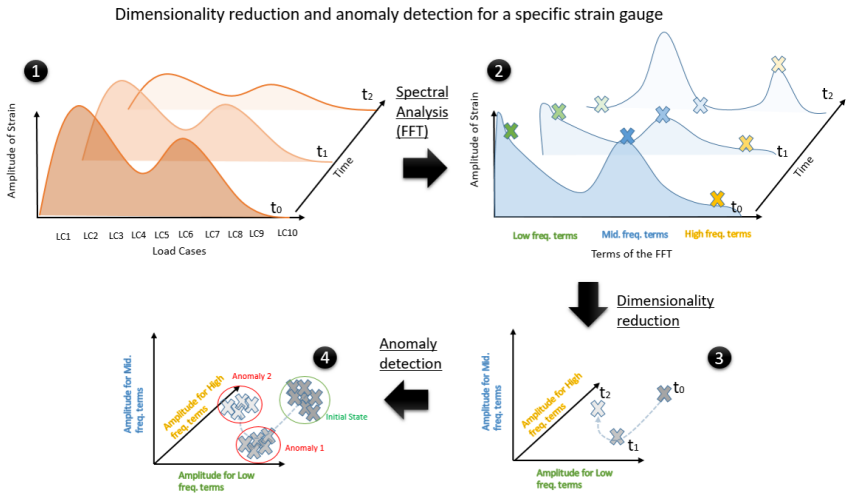


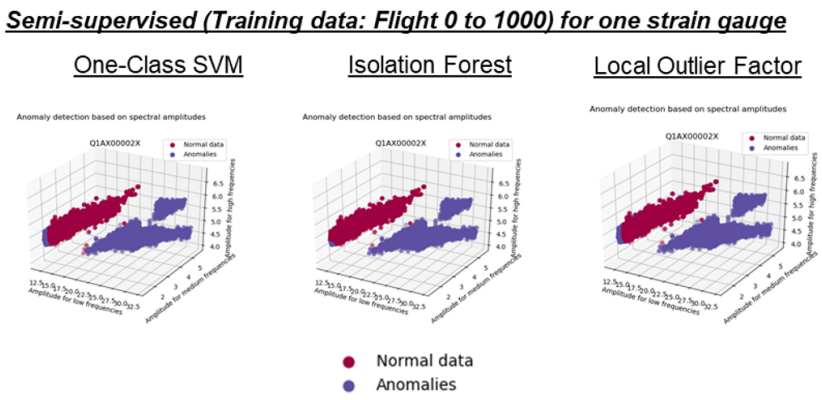

The diameter of each point represents the distance between the point and the centre of the associated cluster (the smaller the point, the further it is from the cluster's centre)
Note: the dataset has been normalized prior to the clustering process

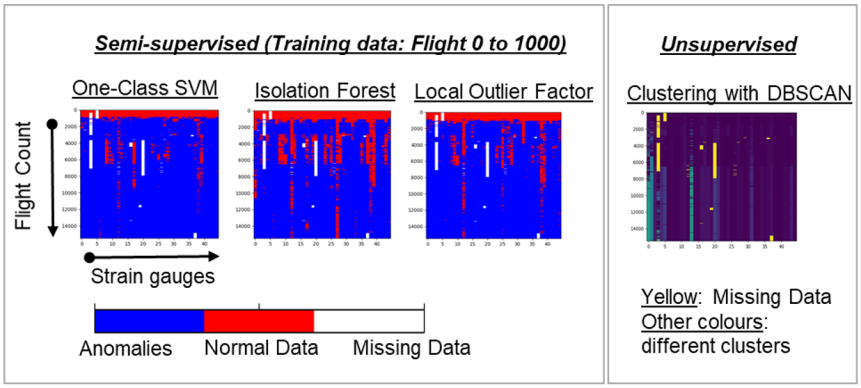


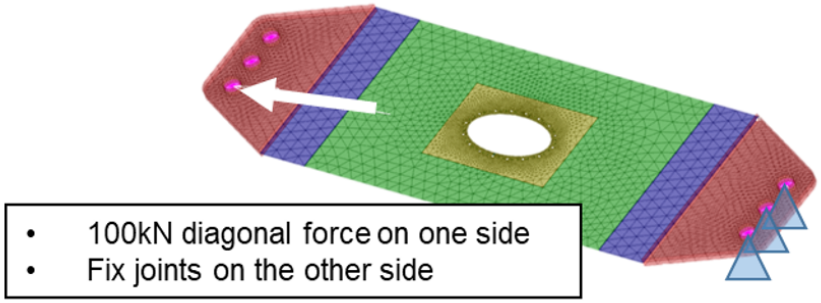



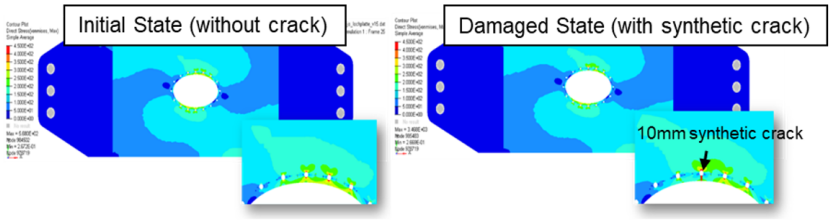



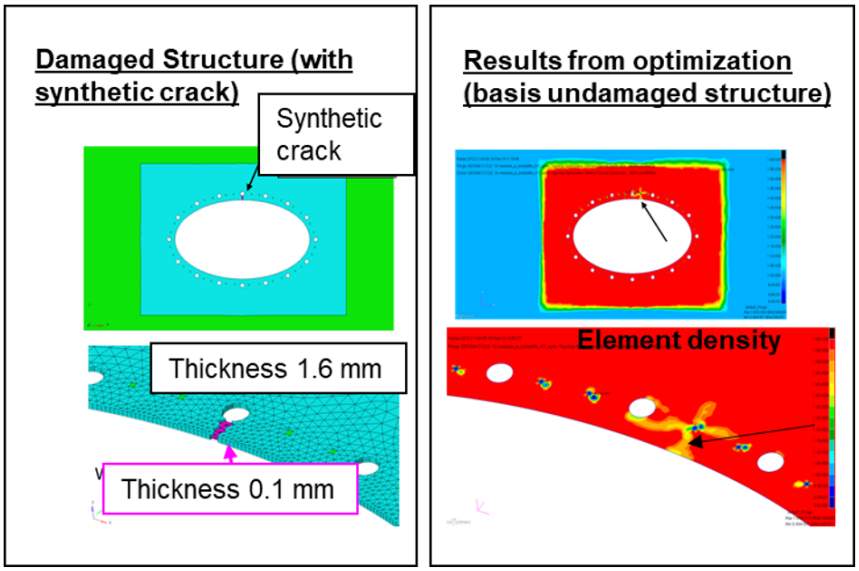





Part of this work was supported by:

Supported by:


on the basis of a decision
by the German Bundestag

This paper is an extended version of the VAL4 conference paper published under the name “Early detection of damages in aircraft structures using Machine Learning and FEM-based methods”.