# Establishing compromise between model accuracy and hardware use for distributed structural health monitoring

**Carolina O. Contente** [*] **Helon V. H. Ayala** [**]

[*] *Departamento de Engenharia Mecânica, Pontifícia Universidade Católica do Rio de Janeiro, RJ, (e-mail: carolinacontente@aluno.puc-rio.br).*
[**] *Departamento de Engenharia Mecânica, Pontifícia Universidade Católica do Rio de Janeiro, RJ, (e-mail: helon@puc-rio.br).*

**Abstract:** Structural health monitoring has been the focus of recent developments in the field of vibration-based assessment and, more recently, in the scope of internet of things as measurement and computation becomes distributed. Data has become abundant even though the transmission is not always feasible at higher frequencies needed for proper assessment, especially in remote applications such as pipelines, subsea, and smart fleets. It is thus important to devise data-driven model workflows that ensure the best compromise between model accuracy for condition assessment and also the computational resources needed for embedded solutions, a topic that has not been widely used in the context of vibration-based measurements. In this context, the present paper proposes a modeling workflow able to reduce the dimension of autoregressive models built on the basis of many acceleration sensors. The three-story building example was used to demonstrate the effectiveness of the method, together with ways assess the best compromise between accuracy and model size. We hope to point future research directions of embedded computing, predictive analytics, and vibration based structural health monitoring, in order to ensure that the models created can be conveniently deployed while optimizing costs for computing infrastructure.

**Resumo**: O monitoramento de integridade estrutural tem sido o foco de desenvolvimentos recentes no campo da avaliação baseada em vibração e, mais recentemente, no escopo da internet das coisas à medida que medição e computação se tornam distribuídas. Os dados se tornaram abundantes, embora a transmissão nem sempre seja viável em frequências mais altas necessárias para uma avaliação adequada, especialmente em aplicações remotas, como dutos, submarinos e frotas inteligentes. Portanto, é importante conceber fluxos de trabalho de modelo orientados por dados que garantam a melhor relação entre a precisão do modelo para avaliação de condição e também os recursos computacionais necessários para soluções incorporadas, um tópico que não tem sido amplamente utilizado no contexto de medições baseadas em vibração. Nesse contexto, o presente trabalho propõe um fluxo de trabalho de modelagem capaz de reduzir a dimensão de modelos autorregressivos construídos com base em diversos sensores de aceleração. O exemplo de construção de três andares foi usado para demonstrar a eficácia do método, juntamente com maneiras de avaliar a melhor relação entre a precisão e o tamanho do modelo. Espera-se apontar futuras direções de pesquisa de computação incorporada, análise preditiva e monitoramento de integridade estrutural com base em vibração, a fim de garantir que os modelos criados possam ser convenientemente implantados enquanto otimiza os custos para a infraestrutura de computação.

*Keywords:* structural health monitoring; autoregressive model; dimensionality reduction; principal component analysis; supervised machine learning; support vector machine; logistic regression; decision trees; random forest; k-nearest neighborhood

*Palavras-chaves:* monitoramento da integridade estrutural; modelo autorregressivo; redução de dimensionalidade; análise de componentes principais; aprendizado de máquina supervisionado; máquina de vetor de suporte; regressão logística; árvores de decisão; random forest; k-nearest neighborhood.

ISSN: 2175-8905

DOI: 10.20906/sbai.v1i1.2767

## 1. INTRODUCTION

Civil, nautical and aeronautical structures, among others, are subject to operational and environmental conditions that may change over time. These changes in operational and environmental conditions impose difficulties in the detection and identification of structural damage Farrar and Worden (2012). Technologies have been developed to replace qualitative visual inspection and time-based maintenance procedures with more quantifiable and automated damage assessment processes. In this context, structural health monitoring (SHM) taken into consideration, which aims to obtain information about the conditions of a given structure or parts of a structure. According to Bornn et al. (2009), there are four steps for SHM: (1) operational evaluation, (2) data acquisition, (3) feature extraction and (4) statistical classification of the features. Some studies have been developed over the years related to data classification using different ways of obtaining features. Figueiredo et al. (2009) studied four different methods for extracting linear features and obtained an optimal linear model for the three-story building problem. Among the methods were Akaike Information Criterion (AIC), Partial Autocorrelation Function (PAF), Root Mean Squared Error (RMSE) and Singular Value Decomposition (SVD); concluding that autoregressive models (AR) of order 5, 15 and, 30 can represent the behavior of the proposed system in a satisfactory way. Pan et al. (2019), in his work, developed a feature extraction method based on Singular Value Decomposition (SVD) by designing a Hankel matrix to enhance multivariate analysis comparing to other traditional feature extraction methods such as autoregressive model (AR) and multivariate vector autoregressive model (VAR). Figueiredo et al. (2010), in another paper, applies machine learning tools to classify the obtained linear features. In this work, four kernel-based algorithms are used to detect damage under varying operational and environmental conditions. Among the methods used are Autoassociative neural network (AANN), Factor Analysis (FA), Mahalanobis Squared Distance (MSD) and SVD; concluding that in terms of general performance, the MSD-based algorithm proved to be the best approach with the lowest type I and II error rates. Pan et al. (2015) applied other different tools for the classification of the features. Methods include One Class Support Vector Machine (One-class SVM), Support Vector Data Description (SVDD), Kernel Principal Component Analysis (KPCA) and Greedy Kernel Principal Component Analysis (GKPCA); concluding that the proposed methods have better classification performance, when compared to methods used in previous works (AANN, FA, MSD and SVD), due to lower classification errors (Type I and Type II). Nguyen et al. (2014) used a method based upon the Monte Carlo simulation methodology to assess the condition of output data, obtained from the autoregressive model. In this work, the order of the autoregressive model is determined by RMSE. Gui et al. (2017) used two types of feature extraction methods: autoregressive model and the residual errors of the statistical parameters. Then grid search method, particle swarm optimization and genetic algorithm were used to determine the parameters in the SVM, which was chosen to perform the classification of the extracted data; concluding that the three methods had good performances although the genetic algorithm based SVM had a better prediction than the others. In the present work is demonstrated not only that the use of AR features of different sensors distributed along the three-story building structure provide the best results in terms of shear accuracy when using the most commonly used shallow models, but also that much smaller models can be obtained when using dimensionality reduction methods before creating the supervised models without sacrificing the model accuracy significantly. More specifically, by using as features the principal components of the feature space composed of the AR parameters of four accelerometers, was observed a decrease in the size of the best model by 27,15%, while the overall accuracy of the model shrank by only 1,64%. It is important to highlight that the size of the models is important as, in general, smaller models tend to generalize better but also, maybe more importantly, smaller models are easier to deploy and to maintain in embedded hardware setups. Additionally, with respect to the use of a single measurement for SHM, we have shown that (i) using sensors closer to the damage location increases the accuracy, and (ii) using more than one sensor, even if it is far from the damage, can increase the accuracy of damage detection. This further highlights the importance of the present paper, as it shows that it is necessary to devise methods that are able to orchestrate many sensors at the same time, while keeping the size of the model compact in order to enable embedded and distributed solutions. In order to assess the results,a Monte Carlo hold-out cross-validation strategy was used. In such strategy, a hundred of models are created by resampling the input-output tuples randomly, for training, validation, and test stages. In this way, the validity of the results is insured in many different realizations of hold-out dataset splitting. A flowchart describing the full process is illustrated in Fig 1.

The paper is organized as follows: Section 2 will introduce the study case used to conduct this research. Section 3 goes through the parameter extraction and dimensionality reduction methods used for this work. Section 4 goes through the machine learning techniques used for data classification and validation strategy. Section 5 shows the results comparing byte size of the models and accuracy results. Conclusions are made in the final section.

## 2. TEST BED AND STRUCTURE

The structure used to carry out this work is illustrated in Fig 2. It consists of a three-story building formed by aluminum plates and columns mounted with bolted joints Figueiredo et al. (2009). This structure was mounted on top of rails in order to allow movement in only one direction (x direction as shown in Figure 1). In addition, a column is arranged in the center of the plate corresponding to the upper floor in order to simulate damage, inducing a non-linear behavior when it collides with a bumper mounted on the floor below. The position of the bumper is adjustable to vary the extent of the impact that occurs at a specific excitation level.

The provided data considered 17 structural states, described in Table 1. The structure was excited ten times for each structural state, to take into account the variability of the data. Thus, for each of the five transducers, ten-time histories were measured for each of the 17 structural

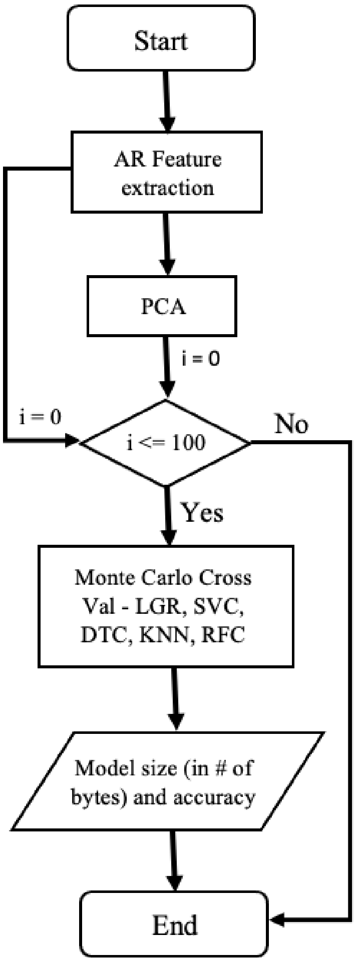
Start
AR Feature extraction
PCA
i = 0
i = 0
i <= 100
No
Yes
Monte Carlo Cross Val - LGR, SVC, DTC, KNN, RFC
Model size (in # of bytes) and accuracy
End

Figure 1. Process flowchart.

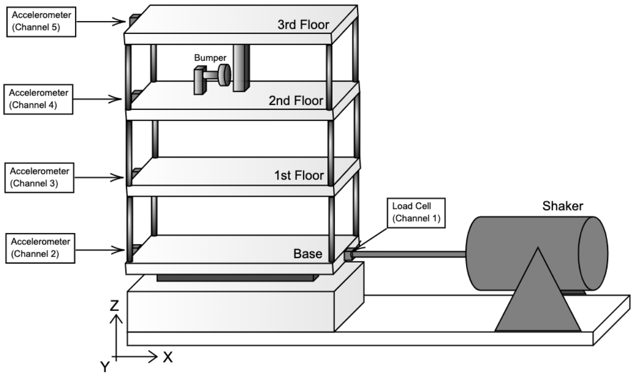
Accelerometer (Channel 5)
Accelerometer (Channel 4)
Accelerometer (Channel 3)
Accelerometer (Channel 2)
3rd Floor
Bumper
2nd Floor
1st Floor
Base
Load Cell (Channel 1)
Shaker
Z
Y
X

Figure 2. Three-story building test bed structure illustration *a* decrescendo.

states (850 tests in total). The data set for the 850 tests performed on the structure consists of an array of rows, columns and depth. The lines refer to all samples obtained during 25 seconds in each of the 850 tests, the 5 columns refer to each of the 5 channels, and the 850 depth columns refers to the 850 tests performed (10 tests for each of the 17 states for each of the 5 channels).

Figure 3 show the acceleration-time history for States 1, 3, 6, 10 and 16. Analyzing the time history, some amplitude differences can be seen as the situation is changing. But it is easier to see the differences and identify any damage that has been put the structure to when looking at the FRF plots or the extracted features of the model, which will be explored in the upcoming sections.

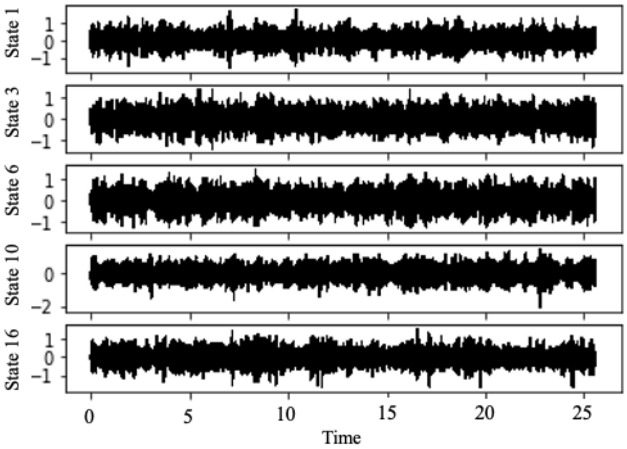

Figure 3. Acceleration-time history from Channel 5 in different states.

## 3. FEATURE EXTRACTION AND DIMENSIONALITY REDUCTION MODELS

The Autoregressive model was chosen for feature extraction since, according to Figueiredo et al. (2009), the AR models can be used as damage-sensitive feature extractor based on the AR parameters (used in this work) or residual errors.

### *3.1 Autoregressive Model (AR)*

To obtain the linear parameters, the autoregressive model AR(p) was considered with a total of $p = n_a$ parameters to estimate, being p the model order, disregarding the input data of the system. It can be written in (1), being $x_i$ the

Table 1. Data labels of the structural state conditions Figueiredo et al. (2009)

| State | Condition | Description |
|---|---|---|
| 1 | Undamaged | Baseline condition |
| 2 | Undamaged | Mass = 1,2kg at the base |
| 3 | Undamaged | Mass = 1,2kg at the 1st floor |
| 4 | Undamaged | 87.5% stiffness reduction in column 1BD |
| 5 | Undamaged | 87.5% stiffness reduction in column 1AD and 1BD |
| 6 | Undamaged | 87.5% stiffness reduction in column 2BD |
| 7 | Undamaged | 87.5% stiffness reduction in column 2AD and 2BD |
| 8 | Undamaged | 87.5% stiffness reduction in column 3BD |
| 9 | Undamaged | 87.5% stiffness reduction in column 3AD and 3BD |
| 10 | Damaged | Gap = 0.20 mm |
| 11 | Damaged | Gap = 0.15 mm |
| 12 | Damaged | Gap = 0.13 mm |
| 13 | Damaged | Gap = 0.10 mm |
| 14 | Damaged | Gap = 0.05 mm |
| 15 | Damaged | Gap = 0.20 mm and 1.2 kg mass at the base |
| 16 | Damaged | Gap = 0.20 mm and 1.2 kg mass at the 1st floor |
| 17 | Damaged | Gap = 0.10 mm and 1.2 kg mass at the 1st floor |

measured signal at time $\mathrm{t_i}$. The $\varepsilon_\mathrm{i}$ term refer to the residual error at the sampling instant i. It can be written as given in (2).

$$x_i = \sum_{j=1}^{p} \phi_j x_{i-j} + \varepsilon_i \tag{1}$$

$$\varepsilon_i = x_1 - \hat{x}_i \tag{2}$$

being $\hat{\mathrm{x}}_\mathrm{i}$ the predicted measure at sampling instant i. The parameter $\phi_\mathrm{j}$ is estimated using batch least-squares approaches or Yule-Walker equations, as stated by Figueiredo et al. (2011).

### *3.2 Principal Component Analysis (PCA)*

According to Jolliffe and Cadima (2016), to interpret large datasets, some methods are required to reduce its dimensionality in an interpretable way preserving most information in the data. One of the oldest methods to do such thing is the principal component analysis (PCA). Mainly, it performs the pre-processing of the data by mean subtraction and setting variance to 1 before performing singular value decomposition (Brunton and Kutz, 2019). It computes the mean matrix as given in (3) where $\bar{\mathrm{x}}$ is the row-wise mean, calculated in (4).

$$\bar{X} = \begin{bmatrix} 1 \\ \dots \\ 1 \end{bmatrix} \bar{x} \tag{3}$$

$$\bar{x}_j = \frac{1}{n} \sum_{i=1}^{n} X_{ij} \tag{4}$$

Subtracting the mean matrix from the large matrix X results in the mean-subtracted data B given in (5).

$$B = X - \bar{X} \tag{5}$$

The first principal component is given

$$u_1 = argmax\ u_1^* B^* B u_1;\ \|u_1\| = 1, \tag{6}$$

being the eigenvector of $\mathrm{B^*B}$ the largest eigenvalue (Brunton and Kutz, 2019).

According to Brunton and Kutz (2019), it is also possible to obtain the principal components by computing the eigenvalue decomposition of the covariance matrix (C), as given in (7) and C is calculated in (8).

$$CV = VD \tag{7}$$

$$C = \frac{1}{n-1} B^* B. \tag{8}$$

Being C the covariance matrix, V the matrix eigenvectors of C and D the diagonal matrix of all eigenvalues of C.

## 4. MACHINE LEARNING SUPERVISED MODELS AND VALIDATION STRATEGIES

In this section, the machine learning methods chosen for this application are presented. All machine learning methods that were chosen to perform the classification of the extracted features are supervised methods and are the following.

### *4.1 Support Vector Classification (SVC)*

Support Vector Machines are a very powerful methods to perform classification of small to medium-sized datasets. It was proposed by Boser et al. (1992). It is a supervised learning algorithm which aims to classify a set of data points that are mapped to a multidimensional characteristic space using a kernel function.

According to Chang and Lin (2011), a training vector in two classes given as $\mathrm{x_i} \in \mathrm{R^n}$, $\mathrm{i} = 1, ..., \mathrm{l}$ and an indicator vector $\mathrm{y} \in \mathrm{R^l}$ as $\mathrm{y_i} \in \{1, 1\}$, SVC solves the primal optimization problem as following

$$\begin{aligned} \min w, b, \xi \frac{1}{2} w^T w + C \sum_{i=1}^{l} \xi_i \\ \text{subject to } y_i \left( w^T \phi\left(x_i\right) + b \right) \geq 1 - \xi_i, \\ \xi_i \geq 0,\ i = 1, ..., l \end{aligned} \tag{9}$$

Where $\mathrm{C} > 0$ is the regularization parameter and $\phi\left(\mathrm{x_i}\right)$ maps $\mathrm{x_i}$ int a higher dimensional space. The main goal is to find $\mathrm{w} \in \mathrm{R^n}$ and $\mathrm{b} \in \mathrm{R}$ so the prediction given by $\mathrm{sign}(\mathrm{w^T}\phi\left(\mathrm{x_i}\right) + \mathrm{b})$ is correct for the majority of the samples. The result for the part $\mathrm{y_i}\left(\mathrm{w^T}\phi\left(\mathrm{x_i}\right) + \mathrm{b}\right))$ is ideally $\geq 1$ for all samples indicating perfect prediction, but not all cases are perfectly separable, so the algorithm allow some samples to be distant in $\xi_\mathrm{i}$ from their correct margin boundary.

The vector variable w can possibly have higher dimensionality and this problem is solved in (10). After the problem solving, the output decision function is given in (11) and its sign correspond to the predicted class.

$$\begin{aligned} \min \propto \frac{1}{2} \propto^T Q \propto -e^T \propto \\ \text{subject to } y^T \propto = 0, \\ 0 \leq \propto_i,\ i = 1, ..., l \end{aligned} \tag{10}$$

$$w = \sum_{i=1}^{l} y_i \propto_i K\left(x_i, x\right) \tag{11}$$

Where e is a vector of ones, Q is a $\mathrm{l} \times \mathrm{l}$ positive semi definite matrix, $\propto_\mathrm{i}$ are the dual coefficients upper-bounded by C and $\mathrm{K}\left(\mathrm{x_i}, \mathrm{x}\right)$ is the kernel given by (12).

$$K\left(x_i, x_j\right) = \phi\left(x_i\right)^T \phi\left(x_j\right) \tag{12}$$

### *4.2 Logistic Regression (LGR)*

Logistic Regression is one of the most common method used for binary data response. According to LaValley (2008), the model takes the natural logarithm of the odds as a regression function of the predictors being the odds the ratio of the probability of the event happening and the probability of the event not happening. It will model the probability based on individual characteristics, which is given by

$$\log\left(\frac{\pi}{1-\pi}\right) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_m x_m \tag{13}$$

where, according to Sperandei (2014), $\pi$ is the probability of the event, $\beta_\mathrm{i}$ is the regression coefficient and $\mathrm{x_i}$ the explanatory variable.

### *4.3 Decision Tree Classifier (DTC)*

According to Agrawal (2014), a decision tree is a structure for express a sequential classification process. According to Rokach and Maimon (2005), a decision tree is formed by nodes that can be "root" nodes that has no incoming edges and "internal or test" nodes that have outgoing edges. All other nodes are called leaves of the tree. Each internal node splits into two or more sub-spaces and each leaf is assigned to one class representing the ideal target value. Some criterions can be used to measure the quality of the tree. Two main methods were used: gini index and information gain. The Gini Index acts measuring divergences between the probability distributions of the target attribute's values and it is defined in (14) (Rokach and Maimon, 2005).

$$Gini\,(y,S) = 1 - \sum_{c_j \epsilon dom(y)} \left(\frac{\left|\sigma_{y=c_j} S\right|}{|S|}\right)^2 \tag{14}$$

Where S is the training set and y is the probability vector of the target attribute. The evaluation criterion for selecting the attribute $\mathrm{a_i}$ is given

$$\begin{aligned} GG\,(a_i,S) = Gini\,(y,S) - \sum_{v_{i,j} \epsilon dom(a_i)} \frac{\left|\sigma_{a_i=v_{i,j}} S\right|}{|S|} \\ \cdot\, Gini\left(y, \sigma_{a_i=v_{i,j}} S\right) \end{aligned} \tag{15}$$

where GG is the Gini Gain. On the other hand, another univariate criterion to decide the best attribute which to split is the impurity-based criterion that uses entropy method as impurity measure.

$$\begin{aligned} IG\,(a_i,S) = E\,(y,S) - \sum_{v_{i,j} \epsilon dom(a_i)} \frac{\left|\sigma_{a_i=v_{i,j}} S\right|}{|S|} \\ \cdot\, E\left(y, \sigma_{a_i=v_{i,j}} S\right) \end{aligned} \tag{16}$$

where E stands for entropy and is calculated in (17).

$$E\,(y,S) = \sum_{c_j \epsilon dom(y)} -\frac{\left|\sigma_{y=c_j} S\right|}{|S|} \cdot\, \log_2 \frac{\left|\sigma_{y=c_j} S\right|}{|S|} \tag{17}$$

The search for a split won't stop until at least one valid partition of the node samples is found.

### *4.4 K-nearest Neighbourhood (KNN)*

The KNN method searches for groups of K objects in the closest training data to similar objects in test data and based on the distance the K nearest neighbors identified and classified (Agrawal, 2014). The Euclidian distance is one common distance metric and is given by (18).

$$d\,(p,q) = \sqrt{\sum (pi - qi)^2} \tag{18}$$

### *4.5 Random Forest Classifier (RFC)*

According to Géron (2017), a random forest classifier is an assembly of several decision trees, generally trained via the bagging method. It creates random decision trees, gets prediction of each tree and selects the best solution by means of voting. In this particular case, the random forest consists in 100 trees and the forest choose a class considering the most out of 100 votes.

### *4.6 Monte Carlo Hold-out Cross Validation*

The validation process is important to guarantee the generalization a machine learning model. For this research, the Monte Carlo Hold-out Cross Validation was chosen. It was proposed by Pan et al. (1984), and, according to Lendasse et al. (2003) in this validation method, the data is randomly divided in several train and validation sets. According to Sperandei (2001), this process is repeated N times (N = 1,2,3, . . . , N) and is defined by (19).

$$MCCV_{n_v}\,(k) = \frac{1}{Nn_v} \sum_{i=1}^{N} \|y_{S_v(i)} - \hat{y}_{S_v(i)}\|^2 \tag{19}$$

where $\mathrm{n_v = n - n_c}$ samples for the validation model, $\mathrm{n_c}$ is the samples for the fitting model, $\mathrm{S_v}$ corresponds to the samples of the validation sets.

## 5. RESULTS AND ANALYSIS

This section will show the results obtained from the dimensionality reduction of the model as well as the classification results of the data. According to Figueiredo et al. (2011), for this particular case, the optimal order stands between 15 to 30. These orders allow discrimination between the undamaged and damaged states when all conditions proposed in Table 1 are considered. For this paper, a model of order 30 is constructed generating, for each channel, one 850 x 31 matrix of parameters. As the features are extracted and plotted in a graph, its clearer to identify and separate the undamaged data from the damage data. Fig. 4 show some of the Channel 5 features plot for undamaged state 1, 3 and 6 and damaged states 10 and 16. It suggests that the more nonlinearities introduced

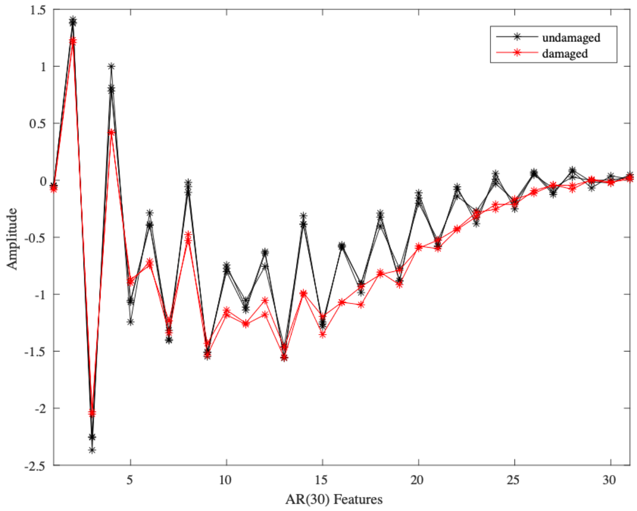

Figure 4. Channel 5 AR(30) features

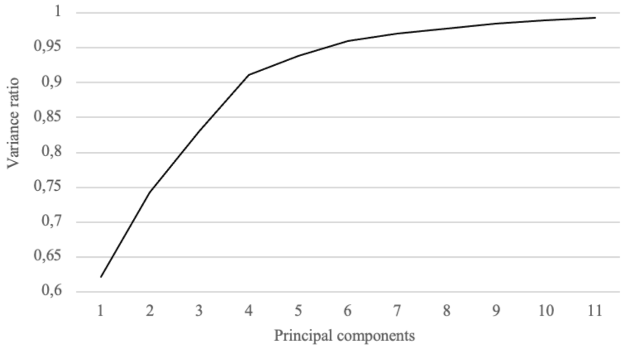

Figure 5. Explained variance ratio vs. total principal components

to the structure, the more decreased is the amplitude of the features. In the upcoming subsections, results of the dimensionality reduction and separation of the data are shown confirming it can be performed with good accuracy comparing to the full data.

### *5.1 Matrix concatenation and model accuracy analysis*

To perform the dimensionality reduction of all concatenated data, the PCA algorithm was designed to retain 99% of the data variability, resulting in a model containing 11 principal components, as illustrated in Fig. 4, resulting in a matrix of 850 x 11 parameters with 99,23% of explained variance. Which means that this smaller matrix in dimensions and byte size can perfectly describe the system and can be well classified using the proposed machine learning techniques.

For the classification step, using Monte Carlo Hold-Out Cross Validation, a hundred models were created randomly for training and test stages from all parameter's matrix obtained during the feature extraction step, using a 50/50 ratio for both the test and training sets. The hyperparameters chosen to perform the classification step are as follows: for LGR, a grid of C values were chosen in a logaritmic scale between $10^{-1}$ and $10^{3}$ and "liblinear" was chosen as the solver; for SVC, the C value grid was the same as LGR. The kernel types used in the algorithm were "linear", "poly", "rbf" and "sigmoid", alternated between iteractions. The degree of the polynomial kernel function was randomly chosen between order two to five and the Kernel coefficient for "rbf", "poly" and "sigmoid" were chosen in a logaritmic scale between $10^{-4}$ and 10; for the DTC model the split criterion used were "gini" for the Gini impurity and "entropy" for the information gain, the split strategy used both "best" and "random" varying in each iteraction; for the KNN model the number of neighbors chosen was a random number between 2 and 100; and finally, for the RFC model, the same hyperparameters used in DTC model were used. Only the number of trees assembled for the classification was set as a random number between 2 and 100.

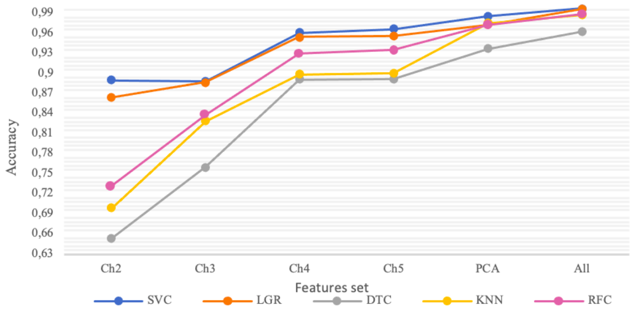

Figure 6. Accuracy plots

Table 2. Accuracy results for every data classification

| | All Feat. | PCA | Ch5 | Ch4 | Ch3 | Ch2 |
|---|---|---|---|---|---|---|
| LGR | **0.9965** | 0.9731 | 0.9563 | 0.9549 | 0.8873 | 0.8644 |
| SVC | **0.9972** | 0.9863 | 0.9664 | 0.9611 | 0.8897 | 0.8882 |
| DTC | **0.9625** | 0.9366 | 0.8915 | 0.8913 | 0.7605 | 0.6540 |
| KNN | **0.9874** | 0.9750 | 0.9002 | 0.8982 | 0.8288 | 0.6986 |
| RFC | **0.9885** | 0.9722 | 0.9354 | 0.9303 | 0.8388 | 0.7317 |

The accuracy results can be seen in Table 2. A better view of the accuracy results is illustrated in Fig. 6. The individual channels classifications perform better from Channel 4 and Channel 5 since their accelerometers are closer to the damage source (bar on the 3rd floor + bumper on the 2nd floor). This pattern follows for all predicted models, where the accuracy results related to the accelerometers that are farther from the damage source are smaller. The accuracy score from the PCA of all data is only 1,64% (medium) less than the full matrix, which can be considerate acceptable since the accuracy remains at good values, above 90%. Nonetheless, the number of inputs is considerably smaller when using PCA if compared to all channels. This will be investigated in the next section.

### *5.2 Model Byte Size Analysis*

When it comes to size in number of bytes of the model, the results of the average number of bytes can be seen in Fig. 7.

The mean values of the sizes were concatenated in Table 3 Putting these data into a graph to better visualize the data, as illustrated in Fig. 8, it is visible that the dimensionality reduction models of the majority of the methods have lower number of bytes size comparing to the individual channels itself. Except for the RFC models

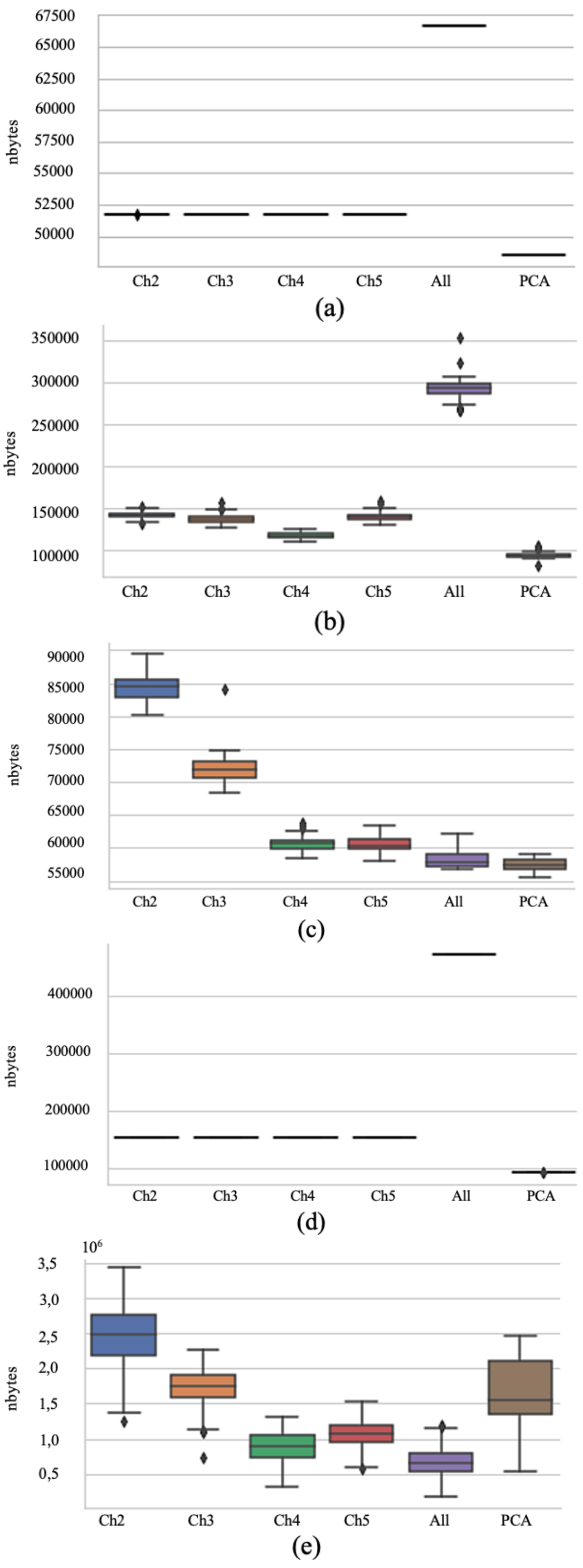

Figure 7. Byte sizes of SVC model (a), LGR model (b), DTC model (c), KNN model (d) and RFC model (e)

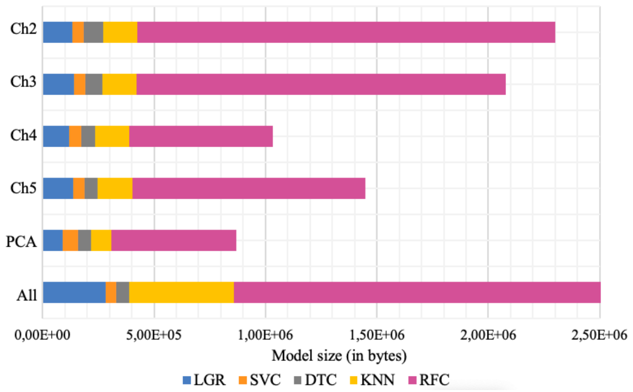

Figure 8. Comparison between model sizes

which are the ones with biggest byte sizes, including the model that had its dimensionality reduced, that has above 2Mb in size. This behavior can be explained since the RFC model is more robust when compared to the other models, requiring more space since, for this case, 100 trees are associated to carry out the classification of the model.

Table 3. Mean bytes numbers result for every data classification model in Mb

| | All Feat. | PCA | Ch5 | Ch4 | Ch3 | Ch2 |
|---|---|---|---|---|---|---|
| LGR | 0,048 | 0,066 | 0,051 | 0,051 | 0,051 | 0,051 |
| SVC | 0,281 | 0,091 | 0,136 | 0,120 | 0,142 | 0,132 |
| DTC | 0,058 | 0,060 | 0,060 | 0,064 | 0,073 | 0,087 |
| KNN | 0,472 | 0,092 | 0,154 | 0,154 | 0,154 | 0,154 |
| RFC | **2,464** | **0,561** | **1,044** | **0,645** | **1,656** | **1,875** |

After performing the dimensionality reduction, the overall byte size of the models is smaller by 27,15% (mean) than all channels concatenated and even the individual channels alone. And, as seen in section 5.1, the accuracy is only 1,64% smaller when comparing the results from all data combined and the PCA data. This can be beneficial in terms of having a smaller data set that describe the behavior of the system properly, which results in a reduction in computational efforts and reduction in the time of execution of the algorithm. Besides, it may cover a majority of cases since larger data sets, like the one in this study case, can eventually have its linear features set reduced to facilitate their deploy.

## 6. CONCLUSION AND FUTURE WORK

Based on linear feature extraction, an approach of dimensionality reduction and several data classification were performed with Monte Carlo hold out cross validation. The main idea is to relate the byte size of the full models and the byte size of the dimensionality reduction of the models, verifying the changes in size and accuracy of the results when applying PCA to the large linear features data set comparing to its original set. A hundred of random validation experiments were conducted and results prove that the dimensionality reduction of this model was well succeeded in terms of size reduction, good description of the model and accuracy results of the classification step, making it reasonable and accurate to work with smaller version models of a bigger data set. Furthermore, results show that the closer from the damage source the data acquisitor is, the better the accuracy results and using more than one source of data can increase these results even if it is far from the damage source. As future work it is intended to extract the nonlinear parameters of this same system to understand the behavior of the results related to the size in bytes of the models and the accuracy of the results.

## ACKNOWLEDGEMENTS

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

## REFERENCES

Agrawal, R. (2014). K-nearest neighbor for uncertain data. *International Journal of Computer Applications*, 105(11), 13–16.

Bornn, L., Farrar, C., and Farinholt, K. (2009). Structural health monitoring with autoregressive support vector machines. *Journal of Vibration and Acoustic*, 131, 021004–1 – 021004–9.

Boser, B., Guyon, I., and Vapnik, V. (1992). A training algorithm for optimal margin classifiers. *Annual Workshop on Computational Learning*, 5, 144 – 152.

Brunton, S. and Kutz, J. (2019). *Data Driven Science Engineering: Machine Learning, Dynamical Systems, and Control*. Cambridge University Press, London.

Chang, C.C. and Lin, C.J. (2011). Libsvm: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3), 1 – 27.

Farrar, C. and Worden, K. (2012). *Structural Health Monitoring: A Machine Learning Perspective*. John Wiley Sons Ltda.

Figueiredo, E., Park, G., Figueiras, J., Farrar, C., and Worden, K. (2009). Structural health monitoring algorithm comparisons using standard data sets. *Los Alamos National Laboratory Report: LA-14393*.

Figueiredo, E., Park, G., Figueiras, J., Farrar, C., and Worden, K. (2010). Machine learning algorithms for damage detection under operational and environmental variability. *Structural Health Monitoring*, 10(6), 559 – 572.

Figueiredo, E., Park, G., Figueiras, J., Farrar, C., and Worden, K. (2011). Influence of the autoregressive model order on damage detection. *Computer-Aided Civil and Infrastructure Engineering*, 26, 225 – 238.

Gui, G., Pan, H., Lin, Z., Li, Y., and Yuan, Z. (2017). Data-driven support vector machine with optimization techniques for structural health monitoring and damage detection. *KSCE Journal of Civil Engineering*, 21(2), 523 – 534.

Géron, A. (2017). *Hands-On Machine Learning with Scikit-Learn and TensorFlow*. O'Reilly Media, Inc., USA.

Jolliffe, I. and Cadima, J. (2016). Principal component analysis: a review and recent developments. *Phil. Trans. R. Soc. A*, 374, 20150202.

LaValley, M. (2008). Logistic regression. *Statistical Primer for Cardiovascular Research*, 117, 2395 – 2399.

Lendasse, A., Wertz, V., and Verleysen, M. (2003). Model selection with cross-validations and bootstraps – application to time series prediction with rbfn models. *ICANN/ICONIP 200, LNCS 2714*, 573 – 580.

Nguyen, T., Chan, T., and Thambiratnam, D. (2014). Controlled monte carlo data generation for statistical damage identification employing mahalanobis squared distance. *Structural Health Monitoring*, 13(4), 461 – 472.

Pan, H., Lin, Z., and Gui, G. (1984). Cross-validation of regression models. *Journal of the American Statistical Association*, 79(387), 575 – 583.

Pan, H., Lin, Z., and Gui, G. (2015). Machine learning algorithms for damage detection: Kernel-based approaches. *Journal of Sound and Vibration*, 363, 584 – 589.

Pan, H., Lin, Z., and Gui, G. (2019). Enabling damage identification of structures using time series–based feature extraction algorithms. *J. Aerosp. Eng.*, 32(3), 04019014–1 – 15.

Rokach, L. and Maimon, O. (2005). *Data Mining and Knowledge Discovery Handbook. Chapter 9 – Decision Trees*. Springer, Boston.

Sperandei, S. (2001). Monte carlo cross validation. *Chemometrics and Intelligent Laboratory Systems*, 56, 1 – 11.

Sperandei, S. (2014). Understanding logistic regression analysis. *Biochemia Medica*, 24(1), 12 – 18.