PAULO CEZAR DIAS JUNIOR
RODOLFO GARCIA NOLETO JUNIOR

REDUCED ORDER MODEL BASED ON MACHINE LEARNING WITH SINDy AUTOENCODERS

FEDERAL UNIVERSITY OF UBELÂNDIA
FACULTY OF MECHANICAL ENGINEERING
2022

PAULO CEZAR DIAS JUNIOR
RODOLFO GARCIA NOLETO JUNIOR

REDUCED ORDER MODEL BASED ON MACHINE LEARNING WITH SINDy AUTOENCODERS

Undergraduate Thesis submitted to the Aeronautical Engineering Graduate Program at the Federal University of Uberlândia, as part of the necessary requirements to obtain the degree of **Bachelor** in **Aeronautical Engineering**.

Concentration Area: Aerolasticity and Loads, Vibrations

Professor Advisor: Dr. Tobias Souza Morais

UBERLÂNDIA - MG
2022

PAULO CEZAR DIAS JUNIOR
RODOLFO GARCIA NOLETO JUNIOR

REDUCED ORDER MODEL BASED ON MACHINE LEARNING WITH SINDy AUTOENCODERS

Undergraduate Thesis **APPROVED** by the Collegiate of the Undergraduate Course in Aeronautical Engineering at Faculty of Mechanical Engineering at Federal University of Uberlândia.

BANCA EXAMINADORA

_____
Prof. Dr. Tobias Souza Morais
Universidade Federal de Uberlândia

_____
Prof. Dr. Daniel Dall'Onder dos Santos
Universidade Federal de Uberlândia

_____
Engenheiro Lucas Portes Leal Ferreira
Empréstimo SIM Santander

UBERLÂNDIA - MG
2022

To God who allows us to
unravel his universe.

NOLETO, R. G; DIAS, P. C. **Reduced Order Models based on Machine Learning with SINDy AUTOENCODERS.** Undergraduate Thesis, Federal University of Uberlandia, Uberlandia, 2022.

Abstract

This work presents a numerical methodology to obtain a reduced order model – ROM – of a linear time-invariant aeroelastic model. The airfoil section of the aeroelastic model, considering only the pitch–plunge motion is studied in a fluid-structure interaction scheme using Computational Fluid Dynamics – CFD. The structural parameters of the linear mechanical system and the necessary data are taken from COMSOL Multiphysics®. In a first analysis, it is shown what is most common in current ROM approaches, a Deep Neural Network - DNN. Such a concept has a high predictive capacity, however, it becomes unfeasible for interpretation and extrapolation of the equations. The Sparse Identification of Nonlinear Dynamics – SINDy – methodology is presented. In order to show the details of the proposed technique, the SINDy technique is implemented in a sparse identification algorithm, Chaotic Lorenz system. Finally, the SINDy methodology is applied to the case of the proposed aeroelastic model. It was possible to identify the system and discover their equations with high accuracy.

Keywords: aeroelasticity, system identification, machine learning, reduced order model, deep neural network, sindy.

NOLETO, R. G; DIAS, P. C. **Reduced Order Models based on Machine Learning with SINDy AUTOENCODERS.** Undergraduate Thesis, Federal University of Uberlandia, Uberlandia, 2022.

Resumo

Este trabalho apresenta uma metodologia numérica para a obtenção de um modelo de ordem reduzida – ROM – de um modelo aeroelástico invariante no tempo. A seção de aerofólio, do modelo aerolástico, considerando apenas os movimentos de translação-rotação, é estudada em um esquema de interação fluido-estrutura usando Dinâmica dos Fluidos Computacional – CFD. Os parâmetros estruturais do sistema mecânico linear e os dados necessários são retirados do COMSOL Multiphysics®. Em uma primeira análise é mostrado o que há de mais comum nas abordagens de ROMs atuais, uma Rede Neural Profunda - DNN. Tal conceito possui elevada capacidade de predição, entretanto, torna-se inviável para interpretação e extrapolação das equações. A metodologia Sparse Identification of Nonlinear Dynamics – SINDy – é apresentada. Inicialmente, com o objetivo de mostrar os detalhes da técnica proposta implementa-se a técnica SINDy em um algoritmo de identificação esparsa, Chaotic Lorenz system. Finalmente, aplica-se a metodologia SINDy para o caso do modelo aerolástico proposto. Foi possível identificar o sistema e descobrir suas equações com alta precisão.

Palavras-chave: Aeroelasticidade, Identificação de sistemas, Machine Learning, Modelo de Ordem Reduzida, Rede Neural Profunda, SINDy.

List of Figures

Figure 2.2.1: Structural representation of the aerolastic system(extracted from BONNAS, MORAIS. 2021).....14
Figure 2.3.1: The architecture of a DNN. The number of hidden layers is optional. (extracted from MathWorks).....17
Figure 2.4.1: Schematic of the SINDy algorithm(extracted from Brunton et. al(2016)). 20
Figure 3.1.1: Loss per number of epochs.....22
Figure 3.1.2: CFD Aerolastic Response and DNN Aerolastic Response at V = 40 m/s 23
Figure 3.1.3: CFD Aerolastic Response and DNN Aerolastic Response at V = 40 m/s 23
Figure 3.2.1: Full Simulation vs Identified System with SINDy.....25
Figure 3.2.2: Numerical derivative and SINDy derivative for ẋ 26
Figure 3.2.3: Numerical derivative and SINDy derivative for ẏ 26
Figure 3.2.4: Numerical derivative and SINDy derivative for ż 27
Figure 3.2.5: Lorenz trajectory and the learned SINDy trajectory for x.....28
Figure 3.2.6: Lorenz trajectory and the learned SINDy trajectory for y.....28
Figure 3.2.7: Lorenz trajectory and the learned SINDy trajectory for z.....29
Figure 3.3.1: Numerical derivative and SINDy derivative for ḣ 31
Figure 3.3.2: Numerical derivative and SINDy derivative for $\dot{\theta}$ 31
Figure 3.3.3: Numerical derivative and SINDy derivative for ḧ 32
Figure 3.3.4: Numerical derivative and SINDy derivative for $\ddot{\theta}$ 32

List of Tables

Table 3.1.1: DNN relevant parameters.....21
Table 3.1.2: DNN results and properties.....24
Table 3.2.1: ROM Chaotic Lorenz System with SINDy results.....29
Table 3.3.1: SINDy relevant parameters.....33

List of Abbreviations and Acronyms

DNN	Deep Neural Network
ML	Machine Learning
SINDy	Sparse Identification of Nonlinear dynamics
ROM	Reduced-Order Model
CFD	Computational Fluid Dynamics
LASSO	Least absolute shrinkage and selection operator
ODE	Ordinary differential equations
STLSQ	Sequentially Thresholded Least Squares
RANS	Reynolds-averaged Navier–Stokes equations
ODEINT	Solve a system of ordinary differential equations
FSI	Fluid-Structure Interaction
TANH	Hiperbolic Tangent

Contents

1 INTRODUCTION....11

2 BACKGROUND THEORY....13

2.1 REDUCED-ORDER MODEL....13

2.2 AEROLASTIC MODEL....13

2.3 DEEP NEURAL NETWORK - DNN....16

2.4 SPARSE IDENTIFICATION OF NONLINEAR DYNAMICS - SINDY....18

3 RESULTS....21

3.1 DEEP NEURAL NETWORK WITH AUTOENCODER....21

3.2 CHAOTIC LORENZ SYSTEM – SINDY METHODOLOGY....24

3.3 AEROLASTIC MODEL IDENTIFICATION WITH SINDY....30

4 CONCLUSIONS AND RECOMMENDATIONS....34

4.1 CONCLUSIONS....34

4.2 RECOMMENDATIONS FOR FUTURE WORK....35

5 REFERENCES....36

1 INTRODUCTION

In the last decades there has been a great demand for the development of tools to facilitate work involving simulations. With the idea of solving this problem, Computational Fluid Dynamics (CFD) emerged. The computational basis of the CFD is a grouping of numerical methods based on the Navier-Stokes equations that help in fluid flow simulations (Kot Engenharia, 2020). From data obtained in CFD simulations it is possible to obtain consistent and valid results. However, CFD still faces many challenges mainly in terms of computational expensiveness and accuracy. With the increasing availability of large amounts of data, data-driven models are starting to be investigated to replace, improve or assist CFD simulations (Calzolari et al, 2021).

Much has been discussed recently about the great evolution of Artificial Intelligence and the capacity for computational simulations and predictions. We saw the emergence of new methods that enabled machines to "learn by themselves" to solve problems and make predictions, such as Machine Learning. The Machine Learning aproach consists in a combination of algorithms and statistical methods, which, upon receipt of a data entry, the algorithm, after analyzing and manipulating them, is capable of generating a data output, which makes it possible to solve problems, forecasts and classifications in different areas, both in the field of medicine, financial market, sports and especially in engineering.

With the evolution discussed above, we have several machine learning methods, in this work, we use Machine Learning and Deep Neural Network, in which both are complementary. As an example, Pure Machine Learning is a system that is capable of making predictions by reproducing data from the past, that is, it uses a wide history of input data to be able to generate output data based on that input, and the Deep Neural Network, where layers of data are added, forming an interconnected network between them, which through this vast database, the algorithm is able to identify patterns, that is, the machine begins to “think alone”, that is, the output data are not based on the input, the algorithm is able to generate the output according to the need, in other words, the data output adapts according to the situation(Brunton et al,(2016)).

Also with the technological evolution, the SINDy method emerged, an

algorithm that allows us to identify linear and/or non-linear dynamics systems, using a one-dimensional system described by:

$$\dot{X} = f(X) \tag{1.1}$$

This system is an illustration of the linear combination of some terms, which we call candidates, with the rate of change of the system's state $\dot{X}$. These candidate terms are functions that can be calculated from the measured data. From this, there was the motivation to carry out this work, where, based on the Machine Learning method combined with the SINDy method, we proved the ability to solve and predict complex problems and identify non-linear models.

Thus, with the help of all these tools described so far, in this work we will show the reduced-order model based on machine learning with SINDy Autoencoders in the next chapters.

2 BACKGROUND THEORY

This chapter presents the numerical methodologies for construction of reduced-order models (ROMs) of an aerolastic system. Firstly, the reduced order model definition is presented, and the aeroelastic model is introduced. Secondly, the DNN concept is presented. Next, ML with SINDy Autoencoders concept and properties are described. Finally, the identification procedure applied to the aeroelastic model is presented in detail.

2.1 REDUCED-ORDER MODEL

The concept of a reduced order model is as simplistic as its purpose is supposed to be: Simplify a more complex model. Reduced-order models must be able to reproduce the main physical aspects of full-scale models, while preserving the expected fidelity within a controlled error. It is convenient to quote:

> A Reduced order model is (ROM) is a simplification of high-fidelity, complex models. They capture the behavior of these source models so that engineers can quickly study a system's dominant effects using minimal computational resources. ROM can be used to simplify various models from full 3D simulations, systems simulations or embedded software. As an added bonus, a ROM ability to simplify complex models means that they can often obfuscate proprietary information. (Ansys, 2019).

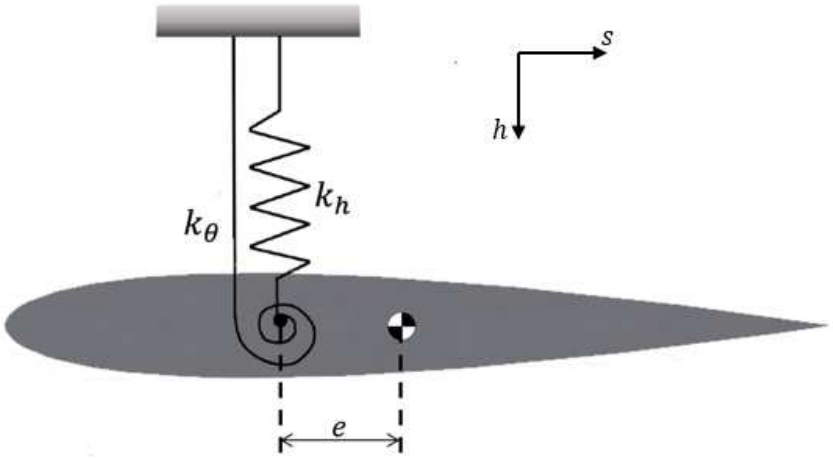
The reduced-order model approach presented in this work is based on the SINDy algorithm.

2.2 AEROLASTIC MODEL

The aeroelastic model was built in COMSOL Multiphysics®. **Figure 2.2.1** presents the structural model of the aeroelastic system, which is based in the cross section of a wing, represented by the NACA 0010 airfoil. The structural model of chord c and wingspan s (considered for mass calculations) has two degrees of freedom (h and θ) with a flexural rigidity k_h and torsional rigidity k_θ. The elastic axis is

defined as the locus of shear centers along a wing (Cooper et al, 2016), clarifying, the elastic axis is the position where only bending happens when the loads is applied on it. In this work the elastic axis is located at a distance *e* ahead the airfoil center of mass.

Figure 2.2.1: Structural representation of the aerolastic system(extracted from BONNAS, MORAIS. 2021)



The aerodynamic model was defined as a turbulent stream in the s direction, conceived by the k - ϵ model. Various forms of the k - ϵ model have been in use for several decades, becoming the most used turbulent model for industrial applications (Spalding, 1974). For the fluid-structure interaction, a fully coupled FSI (Fluid-Structure Interaction) solver was used. This solver computes the couplings that appear at the limits between the fluid and the structure (fluid pressure and viscous forces). For the numerical solution, the Generalized Alpha method was used. The Generalized Alpha method is similar to the Backward Differentiation Formula method (BDF), but differs in its ability to control the degree of damping added by the solver, making the solution more accurate, but less numerically stable (Chung and Hulbert, 1993).

The values for the structural dynamic matrices were obtained with COMSOL®. These matrices can be verified by determining the differential equations of motion of the system using the Lagrange Equation, given by:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_j}\right) - \frac{\partial L}{\partial q_j} = Q_j \qquad (2.1)$$

$$L = T - V \tag{2.2}$$

where T is the kinetic energy, V is the potential energy, q_j are the generalized coordinates and Q_j are the generalized external forces, where *j* is the minimum coordinates number that represents the system (*j* = 1; 2; 3…). The kinetic and potential energy of the structural system are given by Eq. (2.3) and Eq. (2.4), respectively.

$$T = \frac{1}{2}m(\dot{h}(t) + e\dot{\theta}(t))^2 + \frac{1}{2}I\dot{\theta}(t)^2 \tag{2.3}$$

$$V = \frac{1}{2}K_h h(t)^2 + \frac{1}{2}k_\theta \theta(t)^2 \tag{2.4}$$

where *m* is the airfoil mass and *I* is the moment of inertia. Substituting Eq. (2.3) and Eq. (2.4) into Eq.(2.2) and Eq. (2.1) we obtain the equations of motion of the structural model:

$$\begin{bmatrix} m & me \\ me & I+me^2 \end{bmatrix}\begin{Bmatrix} \ddot{h}(t) \\ \ddot{\theta}(t) \end{Bmatrix} + \begin{bmatrix} k_h & 0 \\ 0 & k_\theta \end{bmatrix}\begin{Bmatrix} h(t) \\ \theta(t) \end{Bmatrix} = \begin{Bmatrix} -L(t) \\ T(t) \end{Bmatrix} \tag{2.5}$$

Equation (2.5) can be written as follows:

$$[\boldsymbol{M}]\ddot{\boldsymbol{x}}(t) + [\boldsymbol{K}]\boldsymbol{x}(t) = \boldsymbol{f}(t) \tag{2.6}$$

where ***M*** and ***K*** are the structural mass and stiffness matrices, *L* is the vertical aerodynamic force at the center of pressure, T is the moment caused by *L* in the elastic axis and ***x(t)*** is the state vector. Applying the simplified unstable aerodynamics to the dynamic model in Equation 2.6 it is possible to obtain the expression of the full aeroelastic equation of motion subjected to aerodynamics forces ***f***(t), given by:

$$[\boldsymbol{M}]\ddot{\boldsymbol{x}}(t) + \rho V\left[\boldsymbol{C}_{aero}\right]\dot{\boldsymbol{x}}(t) + \left[\rho V^2 \boldsymbol{K}_{aero} + \boldsymbol{K}\right]\boldsymbol{x}(t) = \boldsymbol{f}(t) \quad (2.7)$$

where $[\boldsymbol{C}_{aero}]$ and $[\boldsymbol{K}_{aero}]$ are the aerodynamic damping and aerodynamic stiffness matrices, ρ is the air density and V is the stream velocity. From the identification methodology, the main objective is to determine the aerodynamic damping $[\boldsymbol{C}_{aero}]$ and the aerodynamic stiffness $[\boldsymbol{K}_{aero}]$ added by the fluid – structure interaction.

2.3 DEEP NEURAL NETWORK - DNN

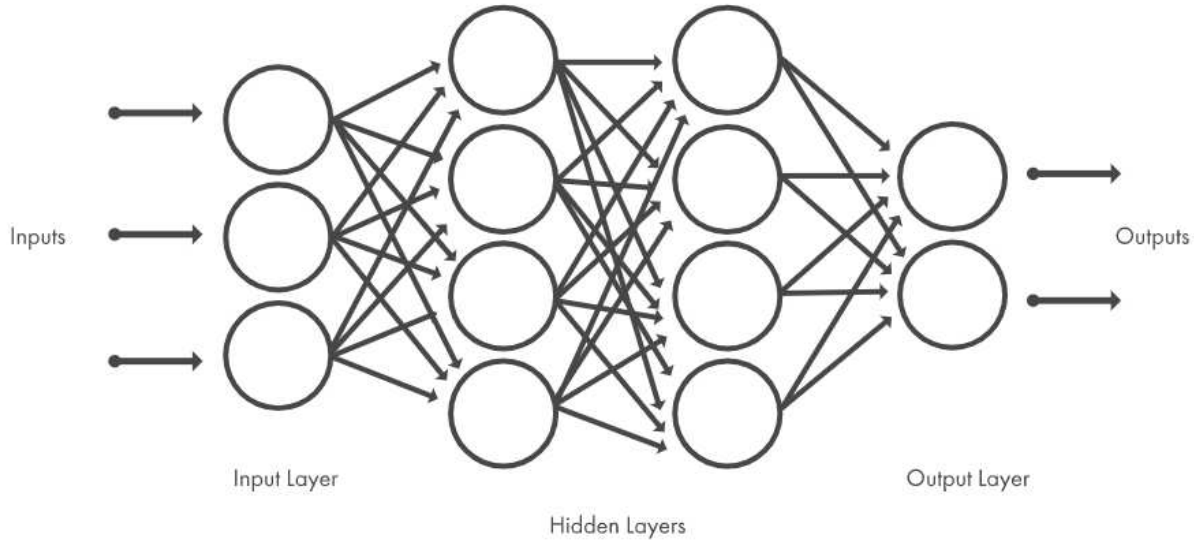
DNN is a kind of Machine Learning that is inspired by the human brain. More precisely:

> The concept of deep neural network is a neural network formed by a combination of a large number of neurons, which simulates the behavior of the human brain to transmit and process information through neuronal connections to respond or solve problems. In the basic architecture of a deep neural network, there are multiple hidden layers between the input layer and the output layer, and each layer is composed of a large number of neurons, and all the neurons in the input layer are individually connected to the neurons in the hidden layer, and the neurons in the hidden layer are also individually connected to the output layer. (Yi-Ren Wang and Yi-Jyun Wang, 2021).

DNN is composed of many artificial neurons, in which each artificial neuron is connected with a weight value, and each artificial neuron has a bias value. The DNN used in this work is an autoencoder type. An autoencoender is a type of multilayer feedfoward neural network that can be used to learn a compressed represetantion of a data. An autoencoder is composed of an with r nodes, followed by a hidden layer with p nodes.

The schematic diagram of a DNN with autoencoder is shown in **Figure 2.3.1**.

Figure 2.3.1: The architecture of a DNN.The number of hidden layers is optional. (extracted from MathWorks)



The neurons (input-output) are connected by the activation functions. The activation functions used in this work are tanh (hiperbolic tangent) and Linear. The Linear and tanh activation functions can be represented as follows:

$$\mathbf{Y_1} = \mathbf{W_2}(\mathbf{W_1}\mathbf{y_1} + \mathbf{b_1}) + \mathbf{b_2} \tag{2.8}$$

$$\mathbf{Y_2} = \mathbf{W_2}\mathbf{tanh}(\mathbf{W_1}\mathbf{y_2} + \mathbf{b_1}) + \mathbf{b_2} \tag{2.9}$$

where $\mathbf{W_1}$ and $\mathbf{W_2}$ are the encoder and decoder weight matrices, and $\mathbf{b_1}$, $\mathbf{b_2}$ are the encoder and decoder biases vectors. Once the optimal { $\mathbf{W_1}$; $\mathbf{W_2}$; $\mathbf{b_1}$; $\mathbf{b_2}$ } are found, we can construct an encoder to reduce the input into a reduced order data using { $\mathbf{W_1}$; $\mathbf{b_1}$ } and a decoder to convert the encoded data back to its original dimension using { $\mathbf{W_2}$; $\mathbf{b_2}$ }(Alyssa Novelia et al. 2021). In a different form, the encoder compresses the input and the decoder attempts to recreate the input from the compressed version provided by the encoder.

2.4 SPARSE IDENTIFICATION OF NONLINEAR DYNAMICS - SINDY

Brunton *et. al (*2016) introduced the SINDy algorithm which identifies ordinary differential equations from data. SINDy follows the assumption that there are only a few important terms that define the dynamics of a system, so that the equations are sparse in the space of candidate functions. Sparse regression is then used to determine the features required to accurately reproduce the system dynamics.

Here, we consider a dynamical system of the form:

$$\frac{dX(\mathrm{t})}{dt} = f(\mathrm{X},\mathrm{t}) \tag{2.10}$$

where X(t) is the state of a system at time t and the function f(X, t) represents the features that define the system dynamics. To determine the function f from data, we collect a time history of state X(t) and its derivative $\dot{X}(t)$. The derivatives $\dot{X}(t)$ can be obtained numerically using the data X(t). The data are stored into two matrices $\boldsymbol{X}$ and $\dot{\boldsymbol{X}}$:

$$\boldsymbol{X} = \begin{bmatrix} X^T(t_1) \\ X^T(t_2) \\ \vdots \\ X^T(t_i) \end{bmatrix} = \begin{bmatrix} X_1(t_1) & X_2(t_1) & \cdots & X_n(t_1) \\ X_1(t_2) & X_2(t_2) & \cdots & X_n(t_2) \\ \vdots & \vdots & \ddots & \vdots \\ X_1(t_i) & X_2(t_i) & \cdots & X_n(t_i) \end{bmatrix} \tag{2.11}$$

$$\dot{\boldsymbol{X}} = \begin{bmatrix} \dot{X}^T(t_1) \\ \dot{X}^T(t_2) \\ \vdots \\ \dot{X}^T(t_i) \end{bmatrix} = \begin{bmatrix} \dot{X}_1(t_1) & \dot{X}_2(t_1) & \cdots & \dot{X}_n(t_1) \\ \dot{X}_1(t_2) & \dot{X}_2(t_2) & \cdots & \dot{X}_n(t_2) \\ \vdots & \vdots & \ddots & \vdots \\ \dot{X}_1(t_i) & \dot{X}_2(t_i) & \cdots & \dot{X}_n(t_i) \end{bmatrix} \tag{2.12}$$

where i is the number of time samples and n the dimension of the system. Next, we construct a library $\boldsymbol{\Theta(X)}$ of candidate non linear functions. Each column of $\boldsymbol{\Theta(X)}$ represents a candidate function for the right-hand side of (2.10). Brunton et. al (2016) suggest that $\boldsymbol{\Theta(X)}$ may consist of constant, polynomial, exponential and trigonometric

terms. The choice of candidate functions normally requires some prior knowledge about the dynamical system. An example of $\boldsymbol{\Theta(X)}$ is shown below:

$$\boldsymbol{\Theta}(\boldsymbol{X}) = \begin{bmatrix} 1 & X_1(t_1) & X_1(t_1)X_2(t_1) & \dots & X_2^2(t_1) & \dots & sin(X_n(t_1)) \\ 1 & X_1(t_2) & X_1(t_2)X_2(t_2) & \dots & X_2^2(t_2) & \dots & sin(X_n(t_2)) \\ \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 1 & X_1(t_i) & X_1(t_i)X_2(t_i) & \dots & X_2^2(t_i) & \dots & sin(X_n(t_i)) \end{bmatrix} \quad (2.13)$$

Now, one can set up a regression problem to determine the unknown parameters.

$$\dot{\boldsymbol{X}} = \boldsymbol{\Theta}(\boldsymbol{X})\boldsymbol{\Xi} \quad (2.14)$$

By using sparse regression, we are able to find what right-hand side terms are small (close to zero), resulting in sparse models. There are a number of algorithms for computing sparse regression. At the present time, the least absolute shrinkage and selection operation (LASSO) (Tibshirani, 1996) is the most popular technique for this type of regression. An alternative algorithm is proposed by Brunton et. al (2016), the sequential thresholded least-squares algorithm. In this algorithm, one starts with a least-squares solution for the unknown parameters and then threshold all parameters that are smaller than some cutoff value.

Figure 2.4.1: Schematic of the SINDy algorithm(extracted from Brunton et al,(2016)).

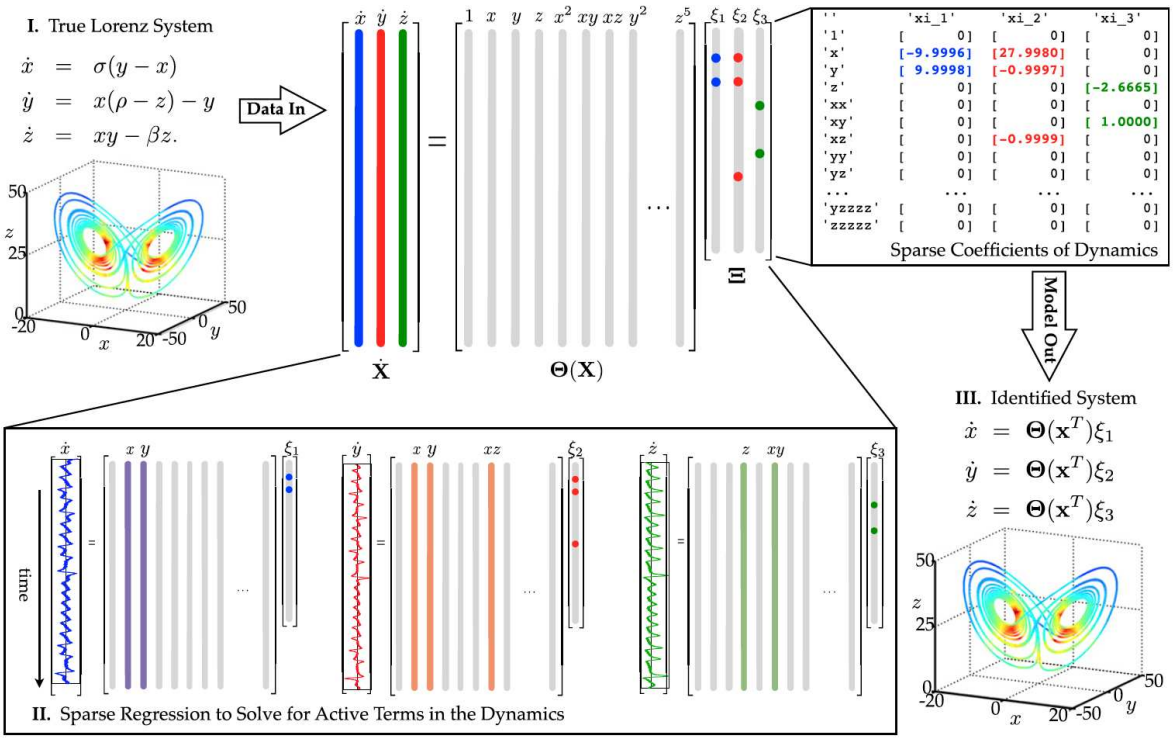


Figure 2.4.1 shows a schematic of the SINDy algorithm demonstrated for the solution of the Lorenz attractor system. First, data are collected, including a time history of the state $\boldsymbol{X}$ and its derivatives $\dot{\boldsymbol{X}}$. Next, a library of candidate functions of the states, $\boldsymbol{X}$, is constructed. Then, sparse regression (fit function) is used to find the fewest terms needed to satisfy Eq. 2.14. After fit the model is possible to see if the identified equations are satisfactory. For that, we compute the derivatives from the model using the predict function in the SINDy. Assume that the model is capable of the reproduce the derivatives with higly precision, we can use the simulate function to evolve initial conditions forward in time using the learned model.

One of the possibilities for the SINDy is work with the scikit-learn library. With that library we be able to use perform a cross-validation using GridSearchCV function. The GridSearchCV function perform a exhaustive search over specified parameter values for an estimator. Then, we can perform some tests searching for the bests parameters before fit the SINDy model.

3 RESULTS

In this chapter we present the obtained results. Recently, several authors have proposed algorithms for the prediction of complex dynamical systems using Deep Neural Network. Jiaqing et al. (2019) used a DNN to capture the dynamic characteristics of aerodynamic and aeroelastic systems for varying flow and structural parameters. Hugo et al. (2019) show that DNN approach is able to learn transient features of a flow. Ling et. al (2016) presented a DNN to improve RANS turbulence models. First, exploring these same methodology we build a DNN with Autoencoder for the aerolastic model. Due to the insufficiency in this model to generate interpretable equations, we focused our efforts on the SINDy methodology. In order to familiarize and facilitate understanding, the SINDy methodology is used to reconstruct a Chaotic Lorenz System. Finally, the obtained results using SINDy for the aerolastic system is presented.

3.1 DEEP NEURAL NETWORK WITH AUTOENCODER

Using the concept of Deep Neural Network, it was possible to obtain satisfactory reconstruction results for the proposed aeroelastic model. For the implementation of this type of Machine Learning it is necessary to choose parameters. The most relevant parameters are presented in **Table 3.1.1.**

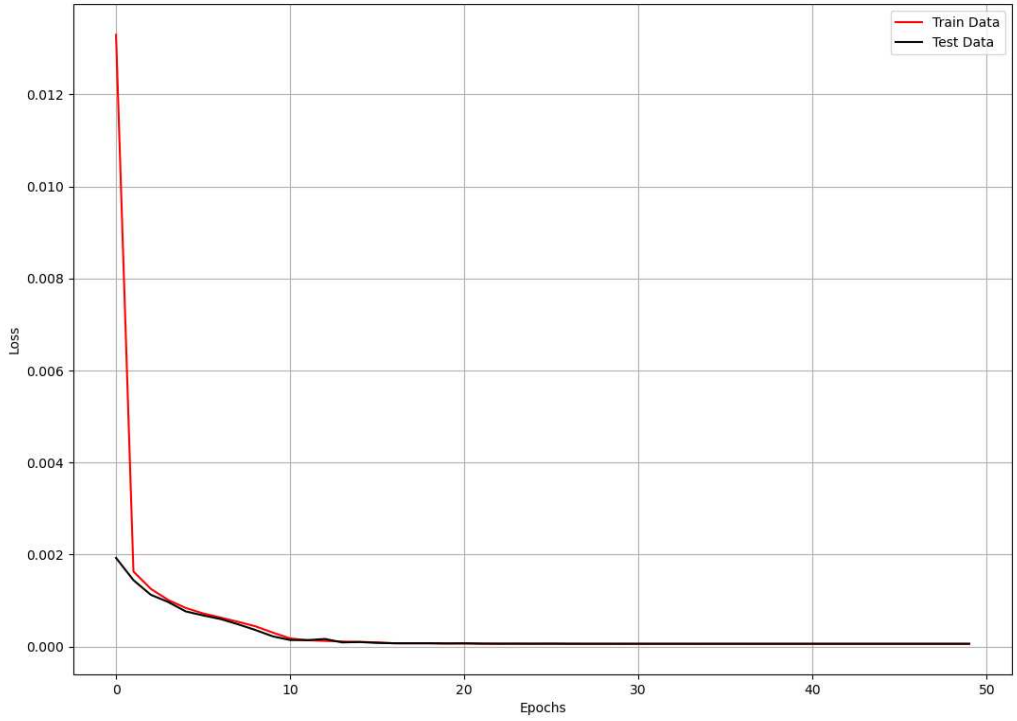
Table 3.1.1: DNN relevant parameters

DNN Autoencoder Architecture	σ_{ENTRY}	σ_{EXIT}	Optimizer	α	λ	η_{ITER}	ϵ
4-2-2	Tanh	Linear	Adam	$1x10^{-3}$	$5.723x10^{-5}$	50	$3.953x10^{-5}$

First, an input with four layers with hyperbolic tangent activation function was considered, after that two layers with linear activation function for the Encoder. The Decoder had two layers with a hyperbolic tangent function at the first output and two layers with a linear function at the second output. In both layers multiple regularizers

were used. We use for the training of the DNN the vertical and rotational displacement data acquired from COMSOL Multiphysics®. Data were divided into a random state. Sixty percent of the data was used for training the DNN. As the optimizer, Adam was chosen. Machine Learning training was done with the no mean squared error metric. **Figure 3.1.1** shows the loss by the number of iterations.

Figure 3.1.1: Loss per number of epochs



The obtained results for the vertical and rotational displacement can be seen in the **Figure 3.1.2** and **Figure 3.1.3**.

Figure 3.1.2: CFD Aerolastic Response and DNN Aerolastic Response at V = 40 m/s

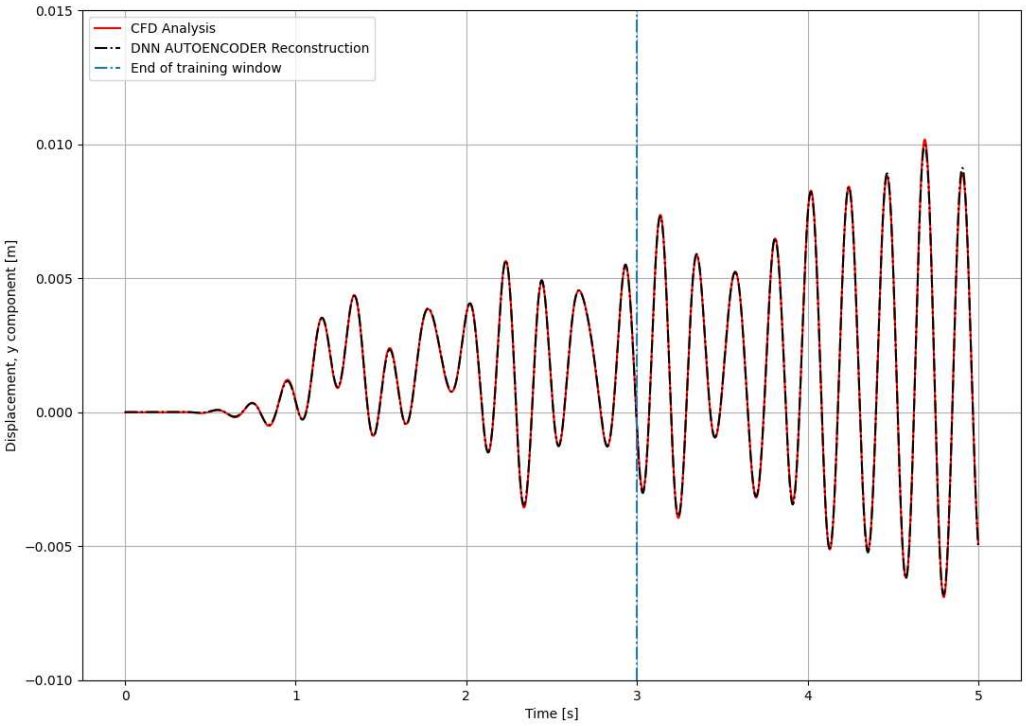
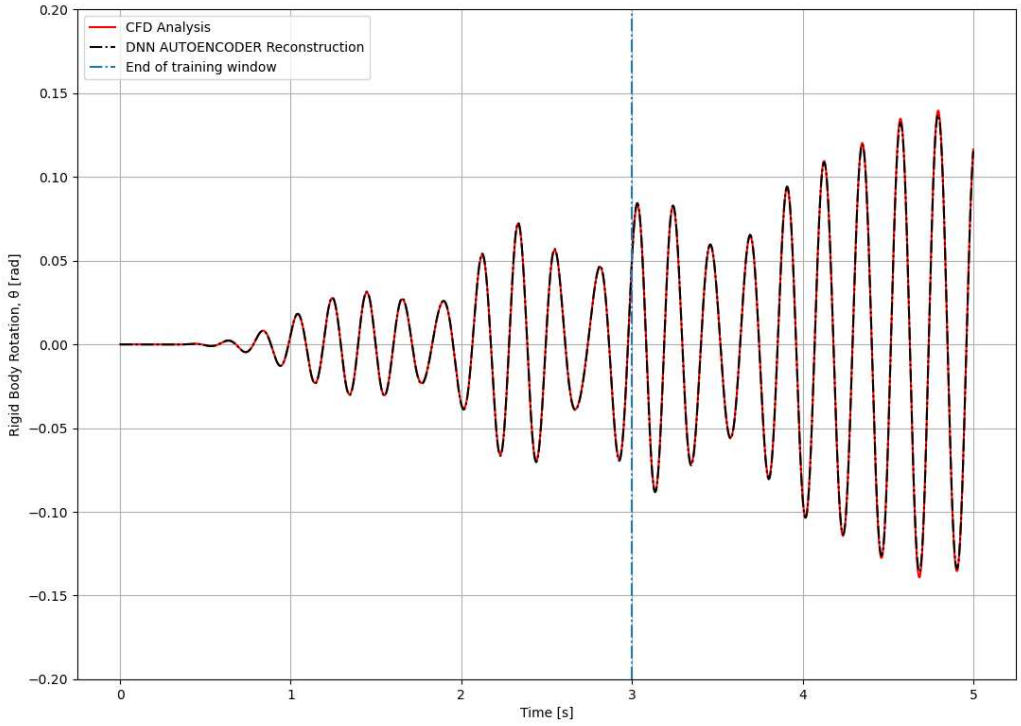


Figure 3.1.3: CFD Aerolastic Response and DNN Aerolastic Response at V = 40 m/s



Results show that the Deep Neural Network with Autoencoder accurately reproduces the aerolastic system. The **Table 3.1.2** shows the accuracy value obtained, the time for training DNN from the data and the specification of the hardware used for this case.

Table 3.1.2: DNN results and properties

DNN Autoencoder	Accuracy	Time [s]	Processor	Memory
	0.9992	144.07	Ryzen 5 5600G	8gb ddr4 4200mhz

The DNN Autoencoder method presents satisfactory results in a shorter time than the COMSOL Multiphysics® software, mentioned in item 2.2, what was already expected whereas “In all of these recent studies, DNN representations have been shown to be more flexible and exhibit higher accuracy than other leading methods on challenging problems” (Brunton and Kutz, 2019, p.220). However, the proposed method does not produce an interpretable equation as a result, being, simply, a linear equation, since this was chosen in the DNN output layer.

3.2 CHAOTIC LORENZ SYSTEM – SINDY METHODOLOGY

For this example, we consider a canonical model in dynamics, the Lorenz system given by:

$$\dot{x} = \sigma(y - x) \tag{3.3}$$

$$\dot{y} = x(\rho - z) - y \tag{3.4}$$

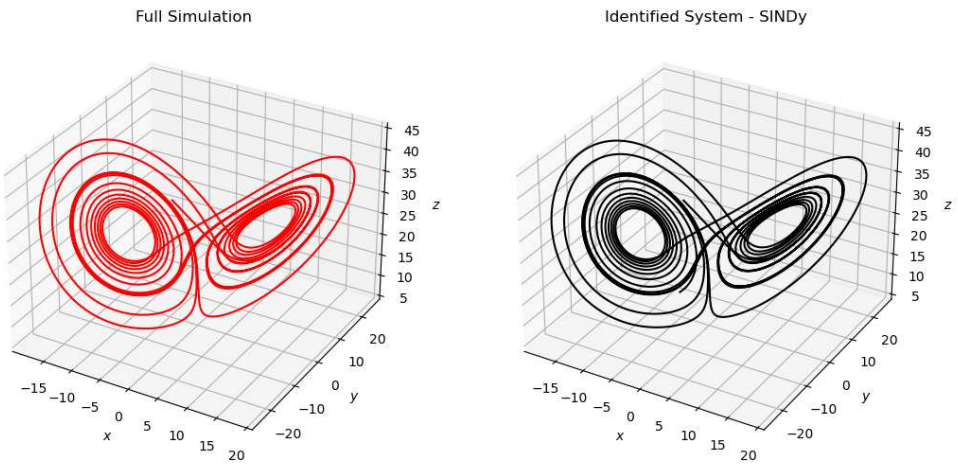
$$\dot{z} = xy - \beta z \tag{3.5}$$

with $\sigma = 10$, $\rho = 28$, and $\beta = \frac{8}{3}$ for this example. The generate training data starting from the initial conditions (-8, 8, 27). using the odeint function in 3.3, 3.4 and 3.5

Equantions. We also consider a time of twenty seconds with a five point zero times ten to the negative five power of timestep. Data are collected and stacked into two large data matrices $\boldsymbol{X}$ and $\dot{\boldsymbol{X}}$, where each row of $\boldsymbol{X}$ is a snapshot of the state $\boldsymbol{X}$ in time, and each row of $\dot{\boldsymbol{X}}$ is a snapshot of the time derivative of the state $\dot{\boldsymbol{X}}$ in time. The main approach is given in **2.4 section**.

Figure 3.2.1 show the measurement data numerically simulated using the Lorenz equations versus the identified system with SINDy methodology for the chaotic Lorenz system.

Figure 3.2.1: Full Simulation vs Identified System with SINDy



Using the predict function is possible to evaluate the derivatives computed by SINDy. **Figure 3.2.2, Figure 3.2.3 and Figure 3.2.4** presents **d**erivatives of variables from the Lorenz equation via numerical differentiation and using a learned SINDy model.

Figure 3.2.2: Numerical derivative and SINDy derivative for $\dot{x}$

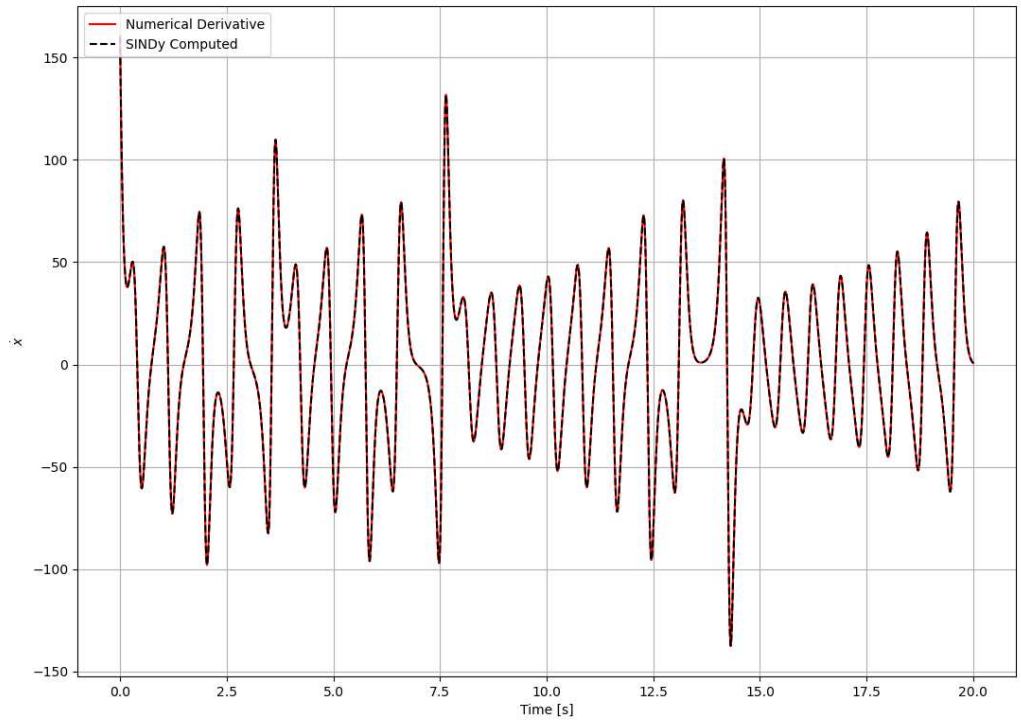


Figure 3.2.3: Numerical derivative and SINDy derivative for $\dot{y}$

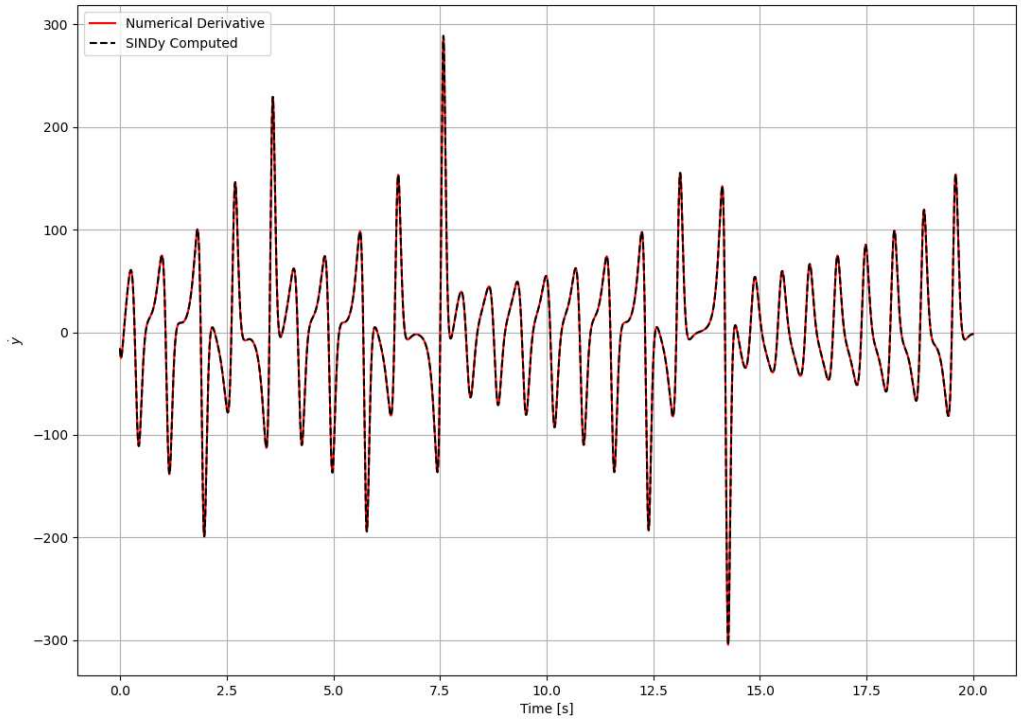
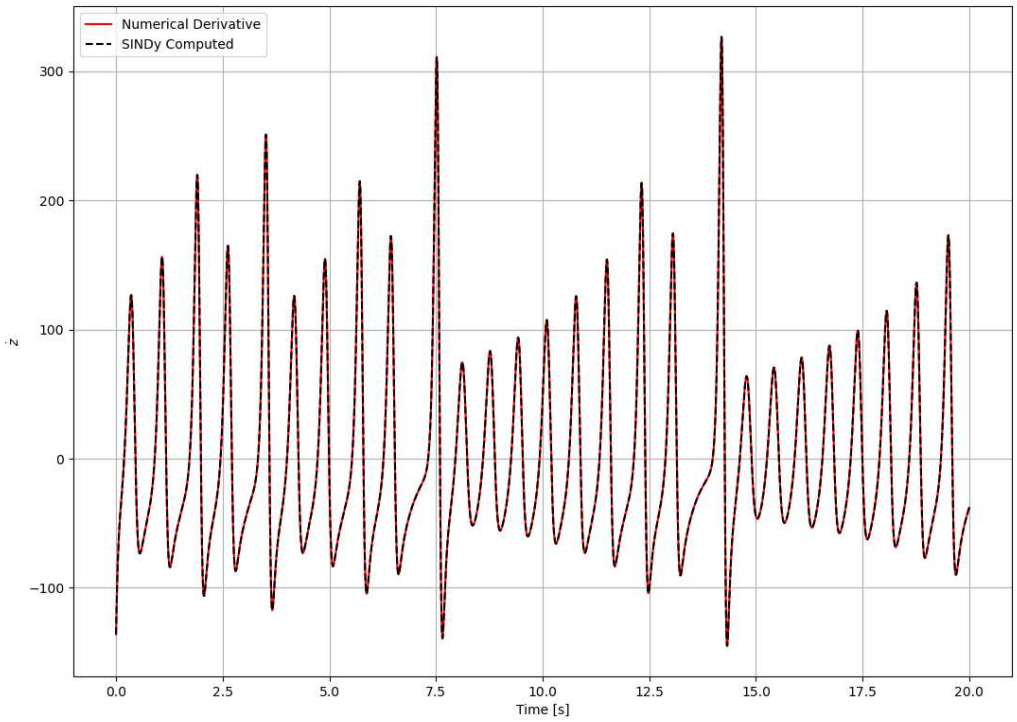


Figure 3.2.4: Numerical derivative and SINDy derivative for $\dot{z}$



Analyzing the derivatives, it is concluded that the system identified by the SINDy methodology reproduces with high precision the chaotic lorenz system.

But, rather than predicting derivatives, we will be interested in using the model to evolve initial conditions forward in time using the learned model. The simulate function does just that.

Figure 3.2.5, **Figure 3.2.6** and **Figure 3.2.7** shows the simulated trajectories against the true trajectories forward in time.

Figure 3.2.5: Lorenz trajectory and the learned SINDy trajectory for x

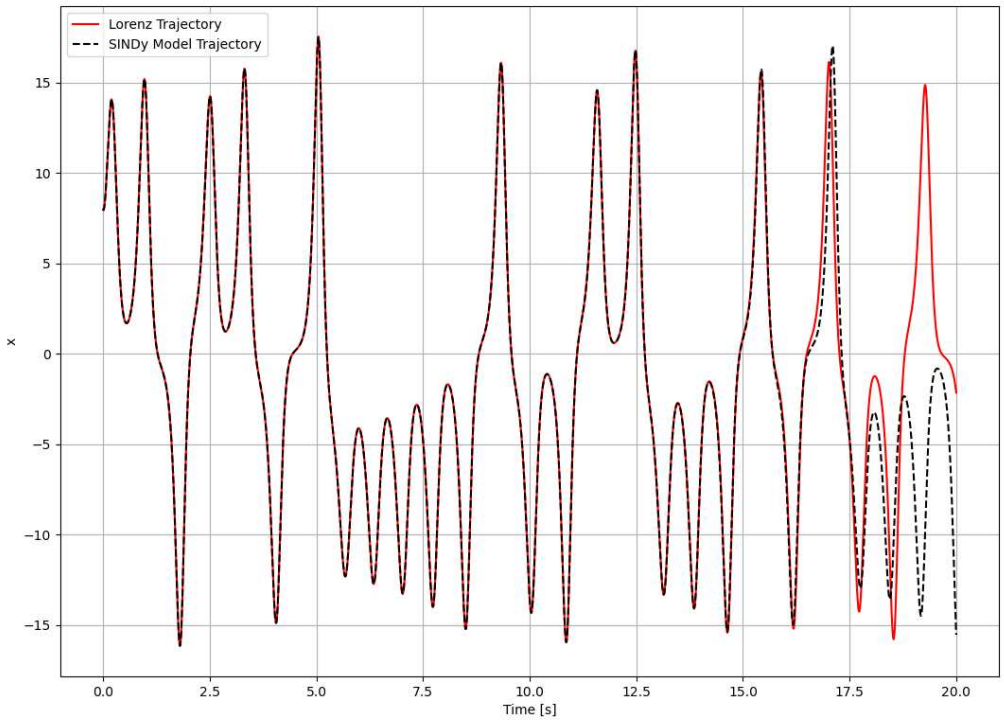


Figure 3.2.6: Lorenz trajectory and the learned SINDy trajectory for y

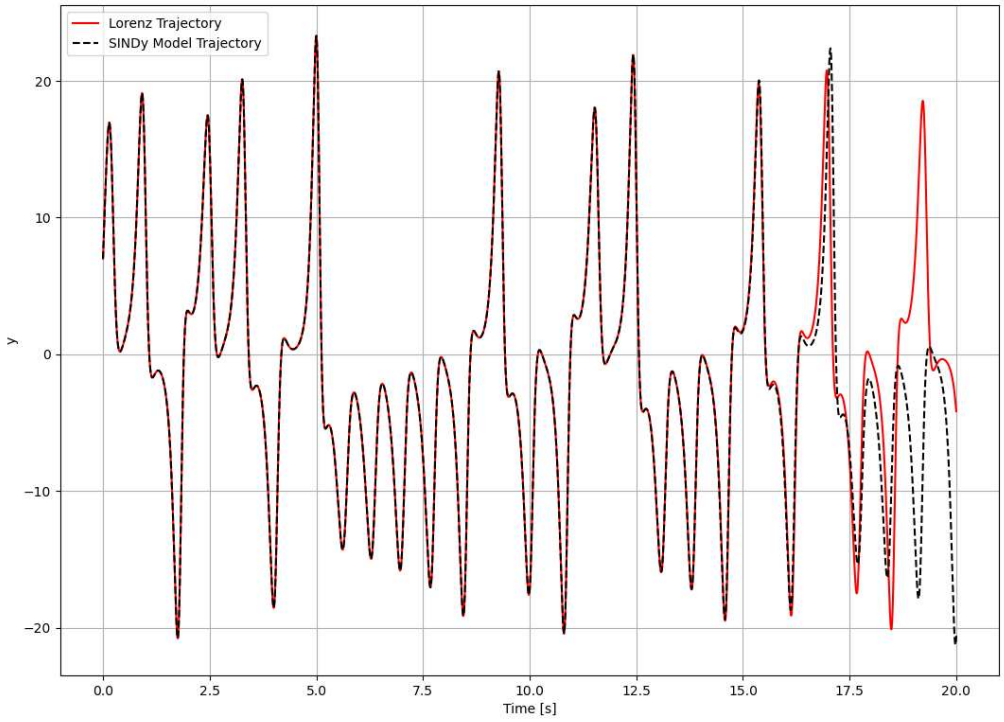
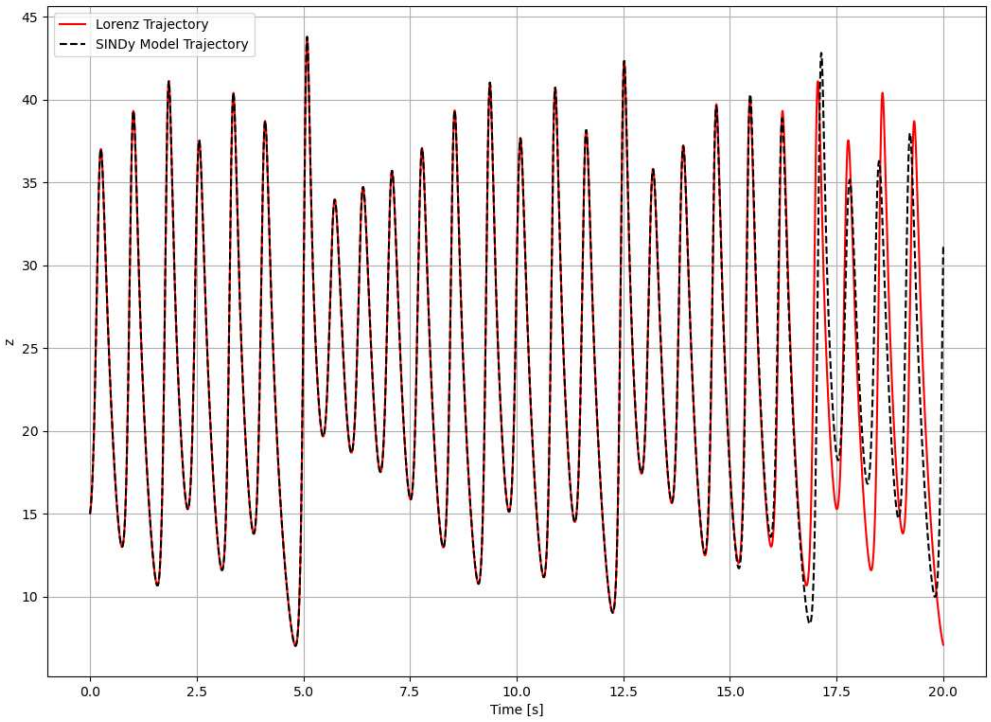


Figure 3.2.7: Lorenz trajectory and the learned SINDy trajectory for z



As we expected, until sixteen seconds (aproximatelly), the trajectories of SINDy agree with the numerical forward in time calculation, but they eventually diverge due to the chaotic nature of the Lorenz equations (Lorenz, 1963). Remembering that a chaotic problem is always a deterministic problem, so, for a known input, random responses arise (SAVI, 1997).

Table 3.2.1 presents the most relevant parameters used by ROM in Charotic Lorenz System with SINDy methodology.

Table 3.2.1: ROM Chaotic Lorenz System with SINDy results

ROM - SINDy	Accuracy	Library	Optimizer
	0.9999	Polynomial (degree 3)	STLSQ (Threshold 0.05)

The derivatives equations obtained can be written in the form:

$$\dot{x} = -10.000x + 10.000y \tag{3.6}$$

$$\dot{y} = 28.000x - 1.000y - 1.000xy \tag{3.7}$$

$$\dot{z} = -2.667z + 1.000xy \tag{3.8}$$

Results obtained from the Equations 3.6, 3.7 and 3.8 shows that the proposed SINDy algorithm accurately reproduces the chaotic Lorenz system dynamics.

Next, we generalize the SINDy method to develop a methodology for construction a reduced-order mode for the proposed aerolastic system in **section 2.2**.

3.3 AEROLASTIC MODEL IDENTIFICATION WITH SINDY

In this case, the aeroelastic model proposed in **section 2.2** is considered. First, it was necessary to process the data. So, we change the data for a state space representation. The vertical and rotational displacement data from COMSOL, along with the values of their first derivatives, were placed in a matrix $\boldsymbol{X}$. The values of the first and second order derivatives, respectively, were placed in a matrix $\dot{\boldsymbol{X}}$. In order to obtain the best accuracy optimizers STLSQ, Lasso and Orthogonal Matching Pursuit were tested. Since the Lasso and Orthogonal Matching Pursuit are sparse optimizers, they presented better results. In the features library were tested the polynomial and Fourier function. The last one presented better results, however, in order to obtain an equation similar to **Equation 2.7**, we opted for the polynomial function.

Figure 3.3.1, **Figure 3.3.2, Figure 3.3.3, Figure 3.3.4** presents derivatives of variables from the aerolastic model via numerical differentiation and using a learned SINDy model computed by the predict function.

Figure 3.3.1: Numerical derivative and SINDy derivative for $\dot{h}$

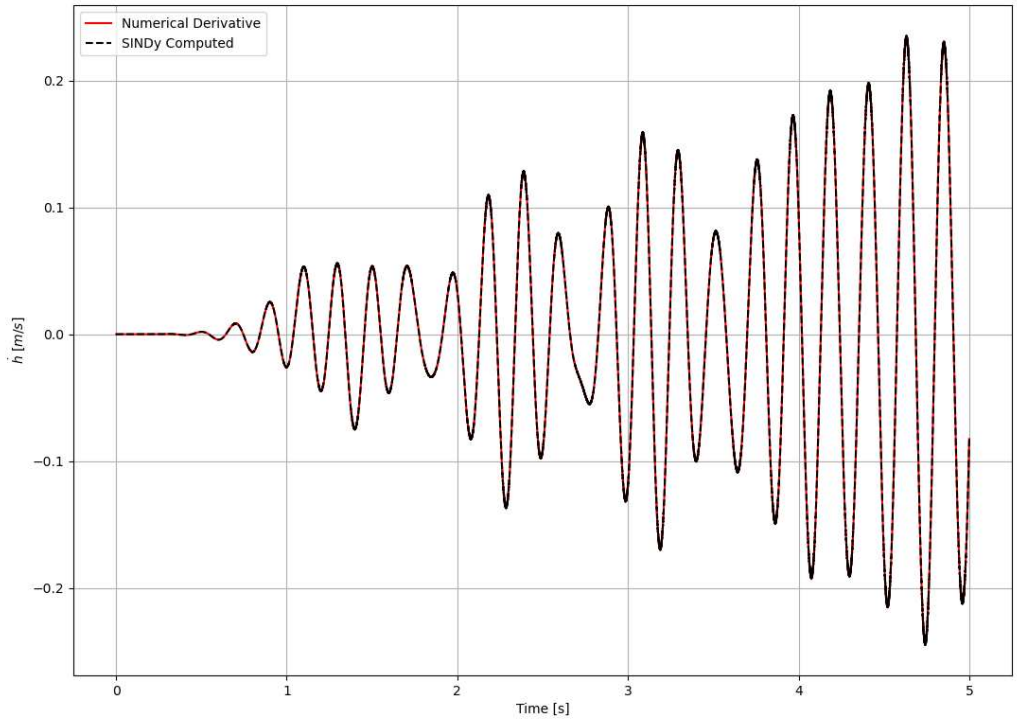


Figure 3.3.2: Numerical derivative and SINDy derivative for $\dot{\theta}$

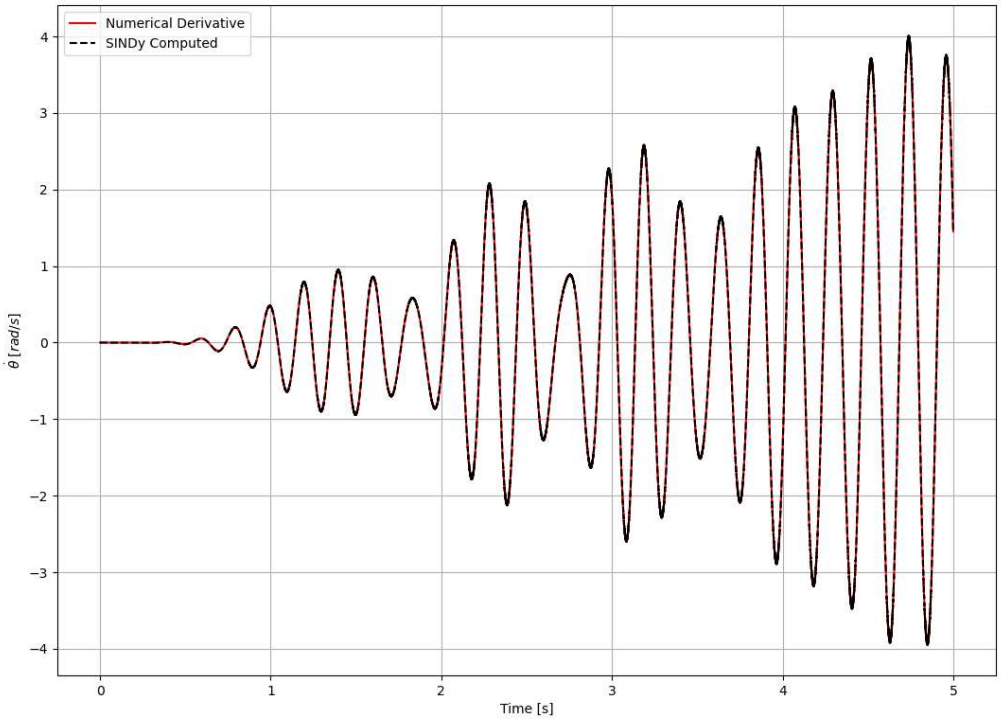


Figure 3.3.3: Numerical derivative and SINDy derivative for $\ddot{h}$

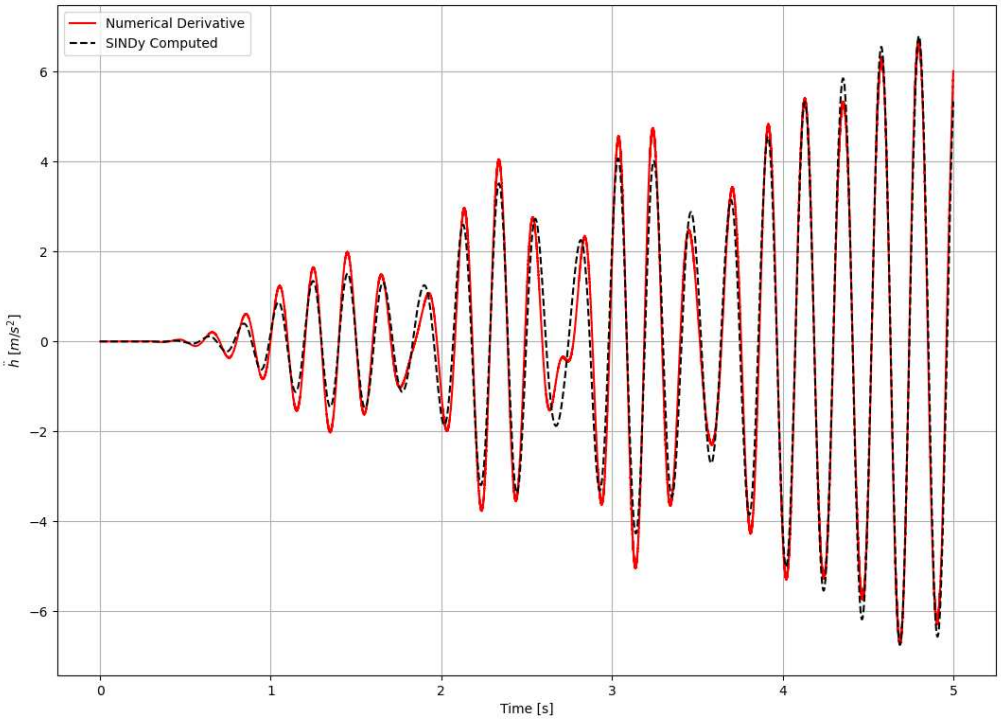
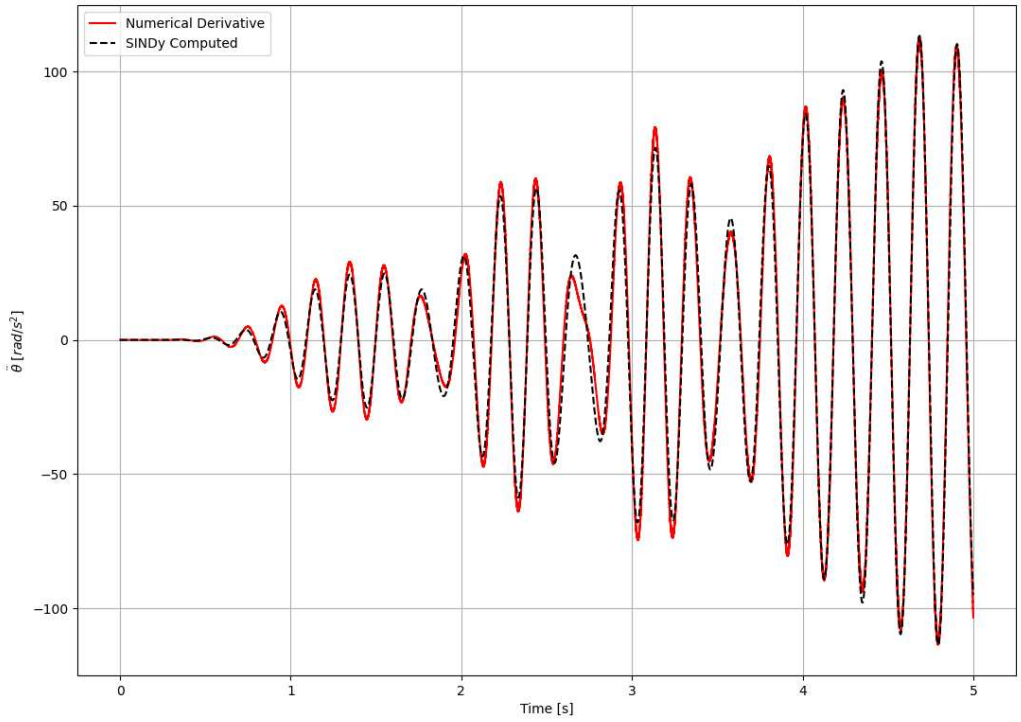


Figure 3.3.4: Numerical derivative and SINDy derivative for $\ddot{\theta}$



These derivatives shows that the SINDy methodology is higly capable of that identify the Aerolastic system.

For the aeroelastic system we tested some parameters in SINDy like the variation of the optimizer (with their intrinsic possibilities) and function libraries. After some tests, a library was used in python – scikit learn. With the help of this library, it was possible to use some tools such as GridSearchCV to find the best parameters to be used.

Table 3.3.1 presents the most relevant parameters used in this methodology for the Aerolastic System proposed found with the scikit learn help.

Table 3.3.1: SINDy relevant parameters

ROM Aerolastic System - SINDy	Derivative Method	Accuracy	Library	Optimizer
	Smoothed Finite Difference	0.9900	Polynomial (degree 3)	Orthogonal Matching Pursuit (coefficients 5)

Then, the ODEs equations obtained can be written in the form:

$$\dot{x} = 1.000\,z \tag{3.9}$$

$$\dot{y} = 1.000\,w \tag{3.10}$$

$$\dot{z} = 48.235 + -0.198\,w + 1.419\,zw + 0.088\,w^2 \tag{3.11}$$

$$\dot{w} = -813.566\,y + 0.618\,w - 4.942\,yw - 33.111\,zw - 1.976\,w^2 \tag{3.12}$$

These results show that the proposed methodology – ROM SINDy – is capable of to identify and represent the proposed Aerolastic model.

4 CONCLUSIONS AND RECOMMENDATIONS

4.1 CONCLUSIONS

We present a methodology for constructing ROM using SINDy. Alternatively, the implementation of a ROM using the DNN concept was shown. The SINDy methodology was implemented following what it was proposed in the literature. In order to facilitate the understanding we present as an example a Chaotic Lorenz system. The details used are described including the algorithms, charts and the equations.

Considering the use of DNN for the aeroelastic model, a high precision was found for the ROM built in a much smaller time interval than those used by current CFD simulators. However, using a DNN framework brings two problems. The first one is that it prevents us from extrapolating the result to other cases, since the resulting equations are just a linear regression with weights and biases, therefore, they are not interpretable. The second is that it was not possible to disentangle or change the training data to other values to predict new results. Still, as it proves to be a highly effective method, we chose to show that results.

Regarding the SINDy model applied to the Lorenz system, it is noted that it is not necessary to do many adaptations. Even so, it was possible to obtain an excellent level of precision for the proposed SINDy method.

Even with the best parameters available for the aerolastic system, it was chosen to use some predetermined parameters. An example is cited in the use of the library. The Fourier library presented better precision, however, with an idea of extending the problem variables and modifying them to the already known ones, we choose the polynomial library. Regarding the chosen optimizer, the optimizers that did not produce a sparse result were abandoned. Thus, the Lasso and Orthogonal Matching Pursuit optimizers were tested to decide which one is more efficient. The Orthogonal Matching Pursuit was chosen due to its capacity in being able to control the amount of terms in final equations.

Unfortunately, due to the method of solution that we choose it was not possible to change the terms of the equations obtained for the known variables such as

velocity, mass and stiffness. We believe that is can solutionated changing the solution mode.

4.2 RECOMMENDATIONS FOR FUTURE WORK

The following improvements could increase the ability of the proposed methodologies to construct ROMs from CFD data.

- To acquire the data for an initial value problem. In the case of DNN or SINDy, acquiring the data as an initial value problem would solve the problem regarding the possibility of not interpreting the data, since, thus, it would only be necessary to change the initial values to obtain new data.

- Improvement in the hyper-parameter optimization process may lead to a speed up in training time and better results. For DNN is equivalent to optimize the values of process like batch size, number of iterations, learning rate and weight decay. In SINDy we can consider another optimizes with their properties.

- Change solution mode. By changing the solution mode, new equations proposed by SINDy are obtained, thus, it may be possible to change the terms of the ROM equation to terms known as velocity, mass and stiffness.

5 REFERENCES

Alyssa Novelia; Yusuf Aydogdu; Thambirajah Ravichandran; Namachvivaya. 2021. Compressed Compressor.

Anderson, John David. 1995. Computational fluid dynamics: the basics with the applications.

Ansys. What is a Reduced Order Model and What's Its Product Development Role?.[2019]. Available:<https://www.ansys.com/blog/what-is-a-reduced-order-model-response-surface-model> Acess: 10 March. 2022.

Bonnas, Pedro; Morais, T., S. 2021. Aerolastic Model Identification of an airfoil with the use of orthogonal functions.

Brunton, S.L.; Proctor, J.L; Kutz, N.J. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. Proceedings of the National Academy of Sciences, vol. 113, n. 15, 3932–3937, 2016. https://doi.org/10.1073/pnas.1517384113

Calzolari, G., & Liu, W. (2021). [Review of Deep learning to replace, improve, or aid CFD analysis in built environment application

Didonna, Marco; Stender, Merten; Papangelo, Antonio; Fontanela, Filipe; Ciavarella, Michele; Hoffmann, Norbert. 2019. Reconstruction of Governing

Equations from Vibration Measurements for Geometrically Nonlinear Systems. Lubricants.7.10.3390/lubricants7080064.

Halder, Rahul; Damodaran, Murali; Khoo, Boo. 2020. Deep Learning Based Reduced Order Model for Airfoil-Gust and Aeroelastic Interaction. AIAA Journal. 58. 1-18. 10.2514/1.J059027. https://doi.org/10.2514/1.J058529

Hugo F. S. Lui; William R. Wolf. 2019. Construction of reduced order models for fluid flos using deep neural networks.

Jadhav, Yayati; Barati, Amir; Farimani. 2021. Dominant motion identification of multi-particle system using deep learning from video.

Kaiser E, Kutz JN, Brunton SL. 2018 Sparse identification of nonlinear dynamics for model predictive control in the low-data limit. Proc. R. Soc. A 474: 20180335. https://doi.org/10.1098/rspa.2018.0335

KotEngenharia. In: What is a Convolutional Neural Network. Acessable in: https://kotengenharia.com.br/analise-numerica/analise-por-cfd/#:~:text=A%20an%C3%A1lise%20fluidodin%C3%A2mica%20computacional20(Computational,simula%C3%A7%C3%A3o%20de%20escoamentos%20de%20fluidos. Access 14 March.2022.

Kou, Jiaqing; Zhang, Weiwei. 2016. Layered reduced-order models for nonlinear aerodynamics and aeroelasticity. Journal of Fluids and Structures. 68. 10.1016/j.jfluidstructs.2016.10.011. https://doi.org/10.1016/j.jfluidstructs.2016.10.011

Li, Kai; Kou, Jiaqing; Zhang, Weiwei. 2019. Deep neural network for unsteady aerodynamic and aeroelastic modeling across multiple Mach numbers. Nonlinear Dynamics. 96. 1-21 10.1007/s11071-019-04915-9. https://doi.org/10.1007/s11071-019-04915-9

Ling, Julia; Kurzawski, Andrew; Templeton, Jeremy. 2016. Reynolds averaged turbulence modelling using deep neural networks with embedded invariance. Journal ofFluidMechanics.807.155166.10.1017/jfm.2016.615. https://doi.org/10.1017/jfm.2016.615

Lorenz, E. N. (1963). Deterministic Nonperiodic Flow, Journal of Atmospheric Sciences, 20(2), 130-141. Retrieved Apr 3, 2022, from https://journals.ametsoc.org/view/journals/atsc/20/2/1520-0469_1963_020_0130_dnf_2_0_co_2.xml. https://doi.org/10.1175/1520-0469(1963)020<0130:DNF>2.0.CO;2

MathWorks. In: What is a Convolutional Neural Network. Acessable in: https://www.mathworks.com/discovery/convolutional-neural-network-matlab.html. Acess: 12 March. 2022.

O. Stodieck, J. E. Cooper, and P. M.Interpretation of Bending/Torsion Coupling for Swept, Nonhomogenous Wings. Weaver Journal of Aircraft 2016 53:4, 892-899. https://doi.org/10.2514/1.C033186

SAVI, M. A. . Caos em Sistemas Mecânicos. Revista Militar de Ciência e Tecnologia , v. XIV, n.4, p. 5-18, 1997.

Silva, Brian; Champion, Kathleen; Quade, Markus; Loiseau, Jean-Christophe; Kutz, J.; Brunton, Steven. (2020). PySINDy: A Python package for the sparse identification of nonlinear dynamical systems from data. Journal of Open Source Software. 5. 2104. 10.21105/joss.02104. https://doi.org/10.21105/joss.02104

Spalding, D., 1974. “The numerical computation of turbulent flow”. Comp. Methods Appl. Mech. Eng., Vol. 3. https://doi.org/10.1016/0045-7825(74)90029-2

Tibshirani, R. Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society. Series B (Methodological), vol. 58, n. 1, 267–288, 1996. https://doi.org/10.1111/j.2517-6161.1996.tb02080.x

Wright, J.R.; Cooper, J.E., 2008. Introduction to aircraft aeroelasticity and loads, Vol. 20. John Wiley & Sons. https://doi.org/10.2514/4.479359