

Additional Graduation Project Report

Understanding The Limitations Of State-Of-The-Art
Nonlinear System Identification Approaches Applied To
Experimental Systems

SAURABH R. MAHAJAN

2022



DELFT UNIVERSITY OF TECHNOLOGY

ADDITIONAL GRADUATION WORK

CIE 5050-09

**Understanding the limitation of state-of-the-art nonlinear system
identification approaches applied to experimental systems**

Author: Saurabh R. Mahajan

ID: (5464315)

Under the supervision of:

Dr. Alice Cicirello

Dr. Luca Marino

November 16, 2022



Abstract

In recent years, there has been a lot of focus on system identification using the vibration data instead of building the mathematical models based on domain knowledge. The Sparse Identification of Nonlinear Dynamical System (SINDy) is a method that has been a great tool to identify the nonlinear dynamics. Therefore, in this report, the nonlinear systems such as duffing oscillator, Single Degree of Freedom (SDoF) with friction is are identified using SINDy algorithm. An impact of noise in the measurement data is studied briefly. The components used in the process of identification are smooth finite difference based differentiator, Sequential Threshold Least Squares (STLSq) optimizer and custom candidate library. Further, a comparison is made between the numerical models and the models obtained from PySINDy. The Root Mean Squares Errors are calculated for all the cases. It is seen that SINDy is capable of identifying these nonlinearities with good accuracy.

Table of Contents

Abstract	a
1. Introduction.....	1
2. Methodology.....	3
2.1. Formulation of SINDy.....	3
2.2. PySINDy package.....	5
2.2.1. Smooth finite difference.....	7
2.2.2. Custom library.....	7
2.2.3. Sequential Threshold Least Squares (STLSq)	7
3. Investigation of equations of motion using PySINDy	7
3.1. Duffing oscillator without noise.....	8
3.2. Duffing oscillator with noise	10
3.3. Numerical friction model without noise	11
3.4. Numerical friction model with noise	13
3.5. Experimental friction model.....	15
4. Quantification of error in equation of motion calculated by PySINDy	17
5. Concluding remarks and future work	18
6. Appendix.....	20
6.1. Appendix A.....	20
6.2. Appendix B.....	22
6.3. Appendix C: Meeting overview	23
7. Bibliography	24

Table of Figures

Figure 1: Schematic of SINDy algorithm for Lorenz system (Taken from [2]).....	4
Figure 2: Components of PySINDy.....	6
Figure 3: Workflow for system identification using PySINDy.....	7
Figure 4: external forcing.....	9
Figure 5: Comparison between PySINDy output and exact numerical system.....	9
Figure 6: External forcing.....	10
Figure 7: Comparison between PySINDy output and exact numerical output.....	10
Figure 8: Single degree of freedom system with friction	11
Figure 9: Base excitation	12
Figure 10: Comparison between PySINDy output and exact numerical output.....	12
Figure 11: Base excitation	13
Figure 12: Comparison between PySINDy output and exact numerical output.....	13
Figure 13: Experimental set up of SDoF system	15
Figure 14: Base excitation	16
Figure 15: Comparison between PySINDy output and exact numerical output (only steady states are compared as we don't have the experimental data for transient response)...	16

Table of Tables

Table 1: Root Mean Square Error (RMSE) of different systems.....	17
Table 2: Absolute error (m) in calculated PySINDy model	20

1. Introduction

The dynamic response of various engineering structures such as buildings, robots, wind turbines etc. depend on behaviour of friction present in the joints. The friction phenomenon can drastically reduce the amplitude of structural response due to dissipation of energy and thus reduce the performance efficiency of the system. The friction has positive impact when it comes to damping of system. The appropriate use of friction force can help in reducing the number of dampers used in the system hence decreasing the expenditure on a system. The presence of friction force makes the system nonlinear in nature. Prediction of response of such nonlinear system is difficult due to lack of a universal and predictive friction model, nonlinear nature of friction forces, the unrepeatable nature of the friction phenomenon [1].

To circumvent this problem the technique of system identification using available response data has gained a lot of attention in recent years. This is possible due to discovery of advanced machine learning algorithms that analyse the data and predict the dynamical system. Various data driven approaches are developed for identifying linear and nonlinear dynamical systems. However, the nonlinear system identification techniques cannot be generalised for all nonlinear systems which creates the need for development in this field.

One of the many derived system identification techniques is System Identification of Nonlinear Dynamical systems (SINDy) introduced in [2], [3] In this report, SINDy is used to identify various systems such as Duffing oscillator without noise, duffing oscillator with noise, Single Degree of Freedom (SDoF) system with friction without noise and with noise, experimental SDoF system having friction contact. The effects of noise levels and complex friction nonlinearity on system identification of duffing oscillator and friction in robotic arm using PySINDy package are studied in reference [4], [5]. However, in this report, the friction nonlinearity is accounted using the Signum function which represents the nonlinearity in more realistic manner and a brief study is done with respect to the noise level.

The duffing oscillator has cubic nonlinearity whereas the SDoF system with friction has friction itself as a nonlinearity. The study in this report focuses on the system identification of nonlinear systems using just the data of forcing and the response. This approach is useful to identify the system in real life scenarios where the response of system is known to a certain excitation. Many times, it is difficult to identify and quantify the type of nonlinearity present

in the structure. Hence using the SINDy algorithm proves to be an effective approach in system identification [6].

2. Methodology

This section focuses on the working principle of SINDy and its components.

2.1. Formulation of SINDy

Dynamical system can be written as:

$$\frac{d}{dt} \mathbf{x}(t) = \mathbf{f}(\mathbf{x}(t)). \quad \text{Equation 1}$$

Where, $\mathbf{x}(t)$ is the state vector denoting states of the system at time t and $\mathbf{f}(\mathbf{x}(t))$ denotes the characteristics of dynamical system such as stiffness, damping, friction force, external forcing on system. The data is collected at various time instants and can be written in following matrix representation [2]:

$$X = \begin{bmatrix} x^T(t_1) \\ x^T(t_2) \\ \vdots \\ \cdot \\ x^T(t_m) \end{bmatrix} = \begin{bmatrix} x_1(t_1) & x_2(t_1) & \cdots & x_n(t_1) \\ x_1(t_2) & x_2(t_2) & \cdots & x_n(t_2) \\ \vdots & \vdots & \ddots & \vdots \\ x_1(t_m) & x_2(t_m) & \cdots & x_n(t_m) \end{bmatrix} \downarrow \text{time} \quad \text{Equation 2}$$

$$\dot{X} = \begin{bmatrix} \dot{x}^T(t_1) \\ \dot{x}^T(t_2) \\ \vdots \\ \cdot \\ \dot{x}^T(t_m) \end{bmatrix} = \begin{bmatrix} \dot{x}_1(t_1) & \dot{x}_2(t_1) & \cdots & \dot{x}_n(t_1) \\ \dot{x}_1(t_2) & \dot{x}_2(t_2) & \cdots & \dot{x}_n(t_2) \\ \vdots & \vdots & \ddots & \vdots \\ \dot{x}_1(t_m) & \dot{x}_2(t_m) & \cdots & \dot{x}_n(t_m) \end{bmatrix} \quad \text{Equation 3}$$

Further, a candidate function library $\Theta(X)$ having various linear, nonlinear functions is built. The candidate library includes user specified functions such as polynomial (with user specified maximum degree), trigonometric functions, exponential functions etc.

$$\Theta(X) = [1 \quad X \quad X^{P2} \quad X^{P3} \quad \dots \quad \sin(X) \quad \cos(X) \quad \dots] \quad \text{Equation 4}$$

Where, the higher order polynomial terms are represented as X^{P2} , X^{P3} , etc, and the X^{P3} can be written as follows:

$$X^{P3} = \begin{bmatrix} x_1^3(t_1) & x_2^3(t_1) & \cdots & x_n^3(t_1) \\ x_1^3(t_2) & x_2^3(t_2) & \cdots & x_n^3(t_2) \\ \vdots & \vdots & \ddots & \vdots \\ x_1^3(t_m) & x_2^3(t_m) & \cdots & x_n^3(t_m) \end{bmatrix} \quad \text{Equation 5}$$

From a point of view of dynamical system, not always all the nonlinearities are active. Hence to add sparsity in the solution or reduce the unnecessary nonlinearities from the solution, a sparse matrix is chosen. This can be also called as selection matrix. It is denoted by $\Xi = [\xi_1 \xi_2 \xi_3 \dots \xi_n]$. Hence the final equation in state space form becomes as follows:

$$\dot{X} = \Theta(X)\Xi \quad \text{Equation 6}$$

For solution of Ξ a regression analysis is done. This will be further discussed in section 2.2.3 in more detail. An effort is made to obtain the equation of motion in the form of velocity which is elaborated in Appendix B. The sample schematic of the SINDy for Lorenz system is shown below for clear understanding:

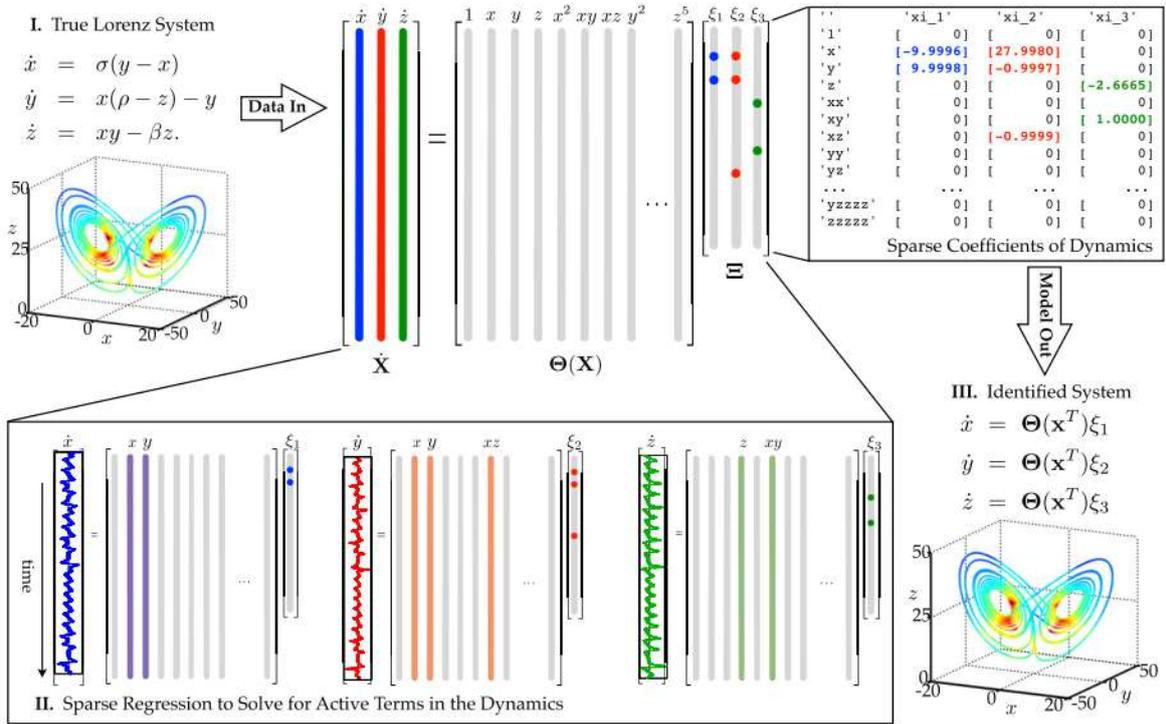


Figure 1: Schematic of SINDy algorithm for Lorenz system (Taken from [2])

In case of dynamical system with external loading (control), the external loading vector and its derivative is added to column of \mathbf{X} and $\dot{\mathbf{X}}$ respectively.

2.2. PySINDy package

For this report, the application of SINDy is done using PySINDy which is a sparse regression package in Python [7], [8]. The PySINDy package consists of components such as differentiators, candidate feature libraries, optimizers, integrators; the schematic of which is shown below:

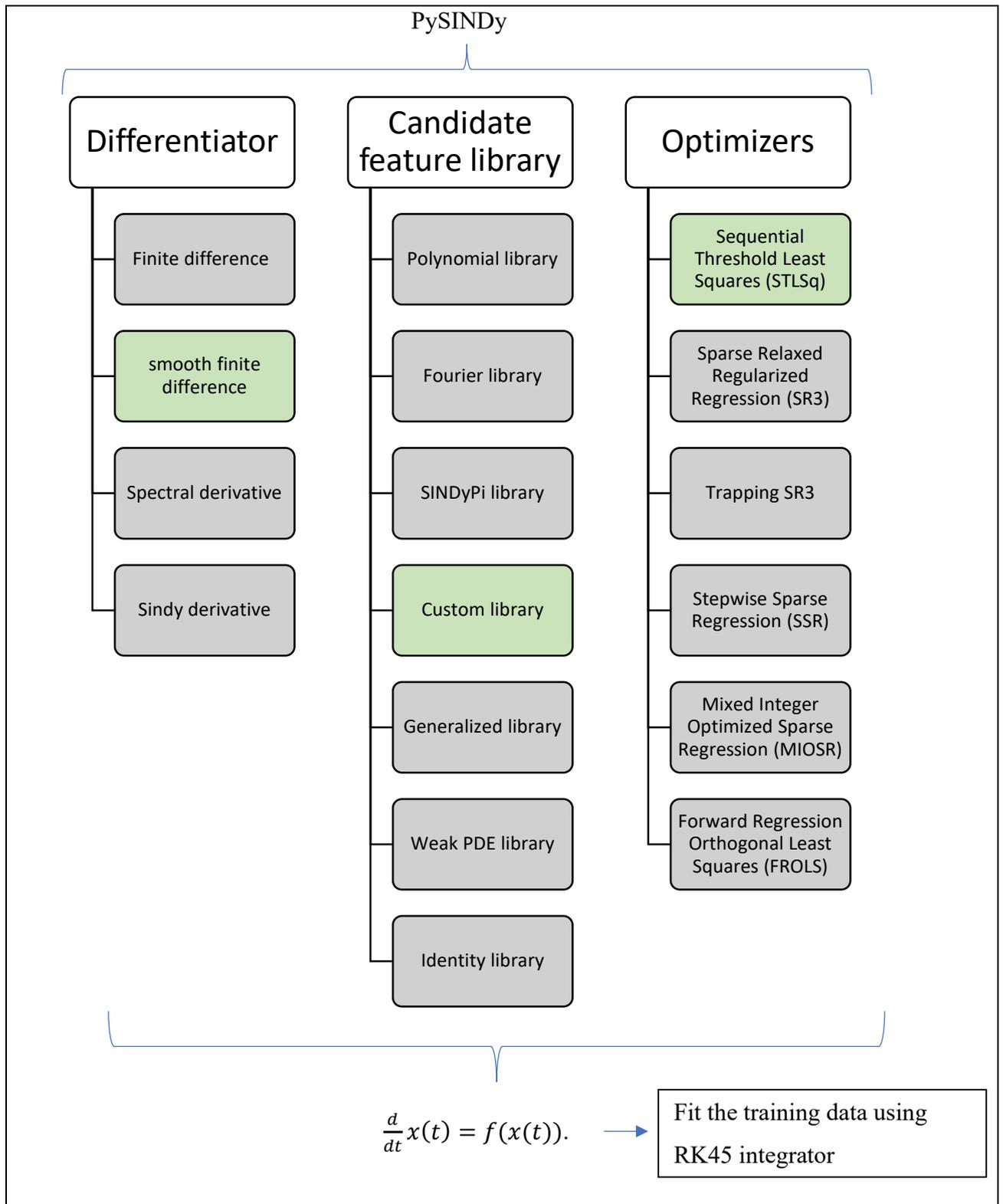


Figure 2: Components of PySINDy

The components marked in green are used for the report.

2.2.1. Smooth finite difference

The method first performs smoothing on the \mathbf{X} data and then calculates finite differentiation to get $\dot{\mathbf{X}}$. The smoothing is done using Savitzky-Golay filter. In this method, the data is divided in small user specified segments and is fitted against a polynomial of user specified order by method of linear least squares.

2.2.2. Custom library

The custom library uses user specified functions i.e., it applies the function on each input variable to, calculate the best fit state space equation. User can specify various functions such as *sin, cos, sign* etc and correspondingly $\sin(\mathbf{X})$, $\cos(\mathbf{X})$, $\text{sign}(\mathbf{X})$ will be used to fit Equation 6

2.2.3. Sequential Threshold Least Squares (STLSq)

$$\|\dot{\mathbf{X}} - \Theta(\mathbf{X})\Xi\|_2^2 + \alpha\|\Xi\|_2^2 \quad \text{Equation 7}$$

The STLSq optimizer minimizes the Equation 7 by using least squares. The user can specify the threshold α or is kept 0.1 by default. The value of α also indicates the sparsity in the solution. The smaller the value, the lesser are the terms we get in the solution.

3. Investigation of equations of motion using PySINDy

The SINDy algorithm is use used for system identification of following systems:

- Duffing oscillator without noise
- Duffing oscillator with noise
- Numerical friction model without noise
- Numerical friction model with noise
- Experimental friction model

General approach of the systems above is given in the schematic below:

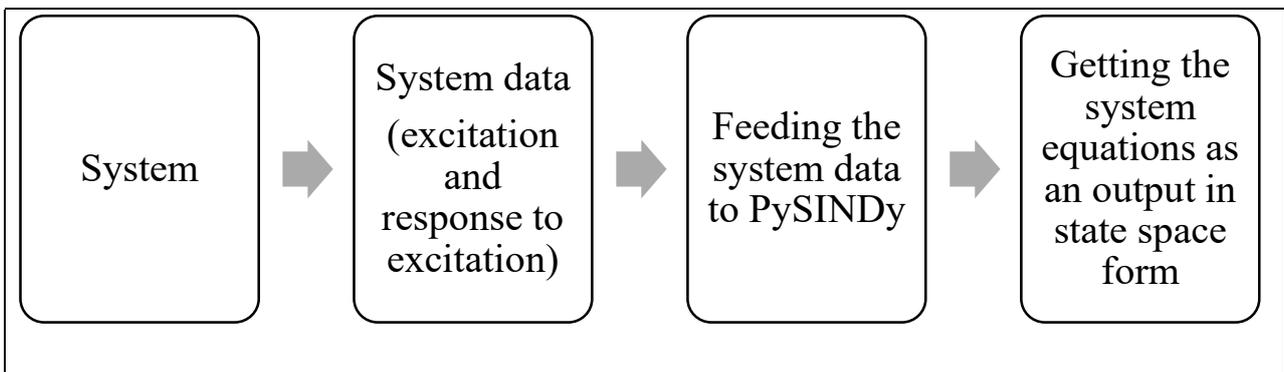


Figure 3: Workflow for system identification using PySINDy

The noise level is decided based on the maximum amplitude of clean x_1 (without noise). The mean is always set to 0 and standard deviation is set to roughly such that $\frac{x_1 \text{ clean}}{\text{standard deviation}} = 10^2$. The noise is added because in reality, there will always be a small error in the measurements and hence an effort is made to check the suitability of PySINDy for real life experimental problems.

3.1. Duffing oscillator without noise

Duffing oscillator has cubic nonlinearity which is comparatively less complex than the friction nonlinearity as there is no sudden jump in the forcing. This forms a good base for verifying the suitability of SINDy algorithm for nonlinear systems [4]. A hypothetical duffing oscillator is represented by following equation of motion:

$$\ddot{x} + 2\zeta\dot{x} + x + \beta x^3 = \Gamma \cos(\rho\tau) \quad \text{Equation 8}$$

Where,

Linear damping coefficient (ζ) = 0.1

Cubic stiffness coefficient (β) = 0.6

Forcing amplitude (Γ) = 1

Frequency of forcing (ρ) = 1.2

Hence the equation of motion can be written in state space form as:

$$\begin{aligned} \dot{x}_1 &= x_2 & \text{Equation 9} \\ \dot{x}_2 &= u - x_1 - 0.6x_1^3 - 0.2x_2 \end{aligned}$$

The external forcing F and displacement x_1 are the 2 inputs for PySINDy. The threshold parameter for STLSq optimizer is 0.15. This parameter should be chosen with great care as it indicates the sparsity of solution [9]. The output equation from PySINDy is as follows:

$$\begin{aligned} \dot{x}_1 &= x_2 & \text{Equation 10} \\ \dot{x}_2 &= 1.077u - 1.051x_1 - 0.595x_1^3 - 0.217x_2 \end{aligned}$$

The testing inputs and output variables are shown below:

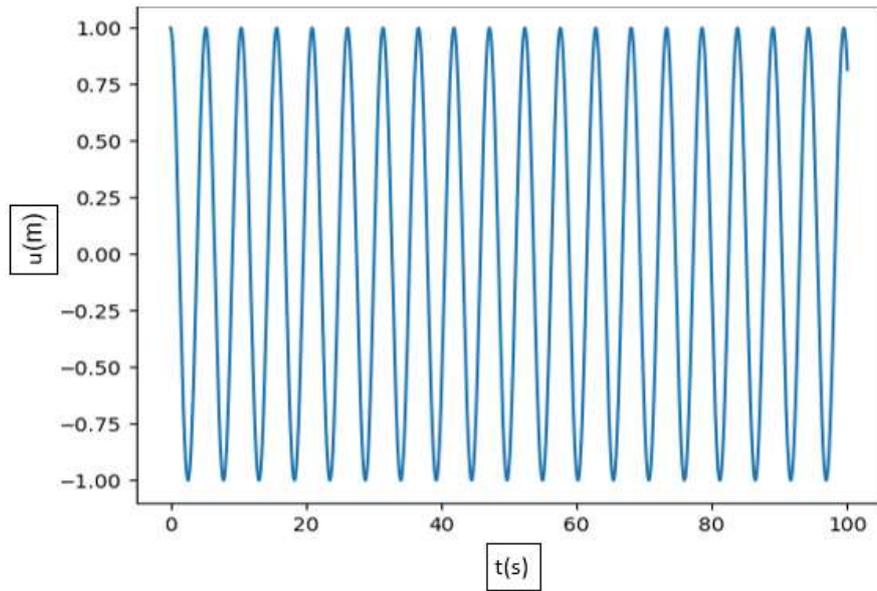


Figure 4: external forcing

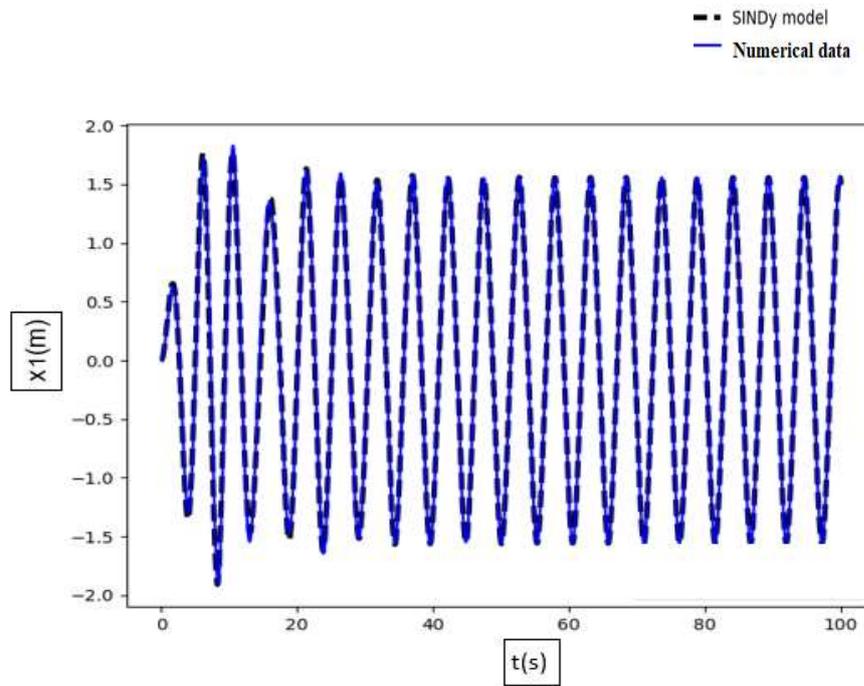


Figure 5: Comparison between PySINDy output and exact numerical system

3.2. Duffing oscillator with noise

The suitability of PySINDy is checked for the noise. The gaussian distributed noise is added to the x_1 state of mean and standard deviation of 0 and 0.01 m, respectively. The only change compared to previous method of duffing oscillator without noise is that now the segment length for smooth finite difference is 81 units. The output equation of motion of PySINDy is as follows:

$$\begin{aligned} \dot{x}_1 &= 0.978x_2 && \text{Equation 11} \\ \dot{x}_2 &= 1.019u - 1.127x_1 - 0.533x_1^3 - 0.2x_2 \end{aligned}$$

The testing inputs and output variables are shown below:

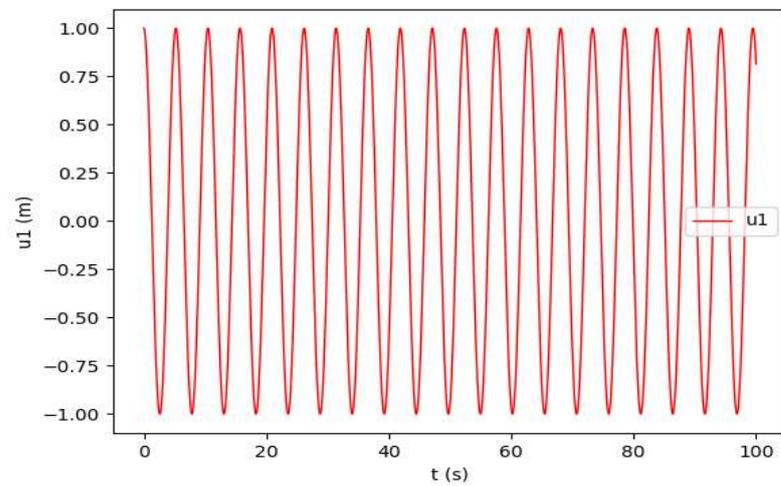


Figure 6: External forcing

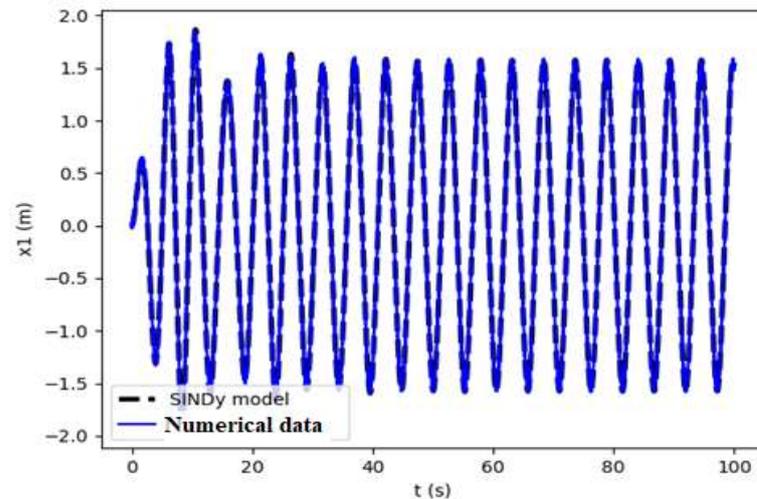


Figure 7: Comparison between PySINDy output and exact numerical output

Hence it is observed that SINDy algorithm works well even in case of noisy measurements which is also found in reference [4].

3.3. Numerical friction model without noise

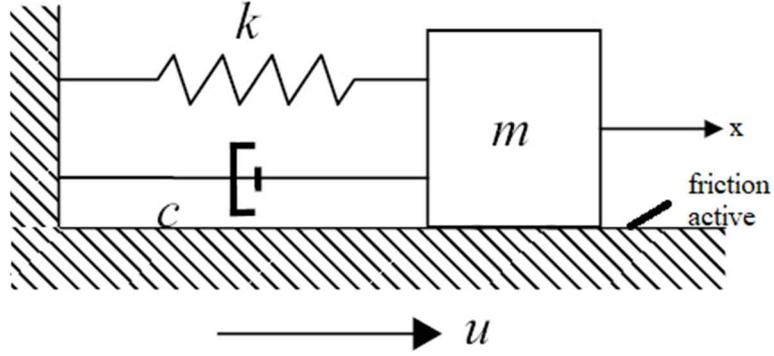


Figure 8: Single degree of freedom system with friction

The considered model is single degree of freedom with Coulomb friction as nonlinearity. The considered motion is continuous due to less friction force acting at the contact. The equation of motion of single degree of freedom with Coulomb friction is as follows:

$$m\ddot{x} + c\dot{x} + kx + F_f \cdot \text{Sgn}(\dot{x}) = ku + c\dot{u} \quad \text{Equation 12}$$

$$x = x_1 \quad \text{Equation 13}$$

$$\therefore \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{c}{m} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ -\frac{F_f \text{Sgn}(\dot{x})}{m} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{ku + c\dot{u}}{m} \end{bmatrix}$$

Where, k , c , m , F_f , u are the stiffness, damping coefficient, mass, magnitude of friction force and base excitation, respectively. The actual equation of motion considering the values of all the parameter is

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -358.706 & -0.0658 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ -0.0856 \text{Sgn}(\dot{x}) \end{bmatrix} + \begin{bmatrix} 0 \\ 358.706u + 0.0658\dot{u} \end{bmatrix} \quad \text{Equation 14}$$

Due to considerable nonlinearity, PySINDy is not able to calculate the equation with great accuracy. To circumvent this issue, a group of data is given as input to PySINDy. The u with driving frequencies 2.013 Hz, 2.583 Hz, 4.611 Hz, 5.242 Hz and corresponding x_1 are taken as inputs for PySINDy. The output equation of motion of PySINDy is as follows:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -358.849 & -0.088 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ -0.085 \text{Sgn}(\dot{x}) \end{bmatrix} + \begin{bmatrix} 0 \\ 358.425u + 0.08\dot{u} \end{bmatrix} \quad \text{Equation 15}$$

The following are the testing input variables u (driving frequency 5.843 Hz), x_1 and corresponding PySINDy output:

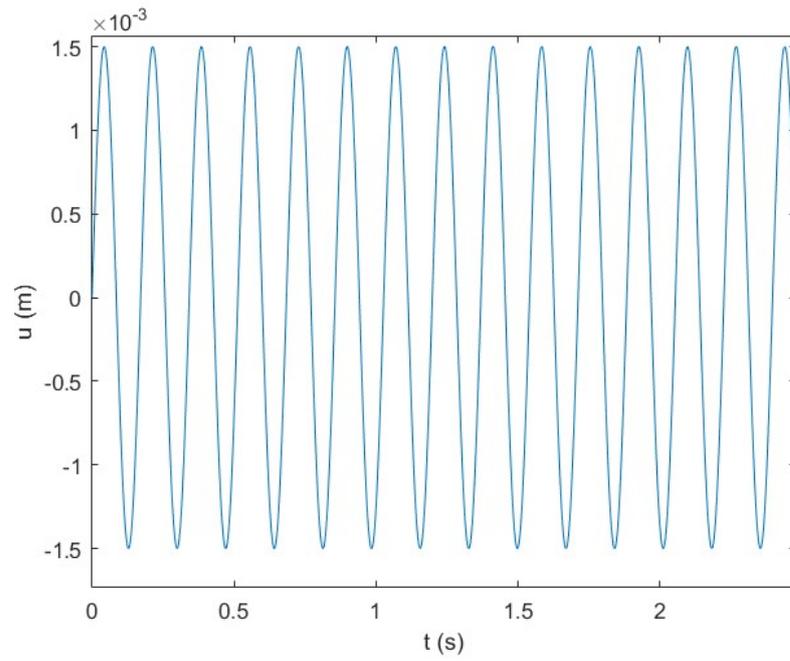


Figure 9: Base excitation

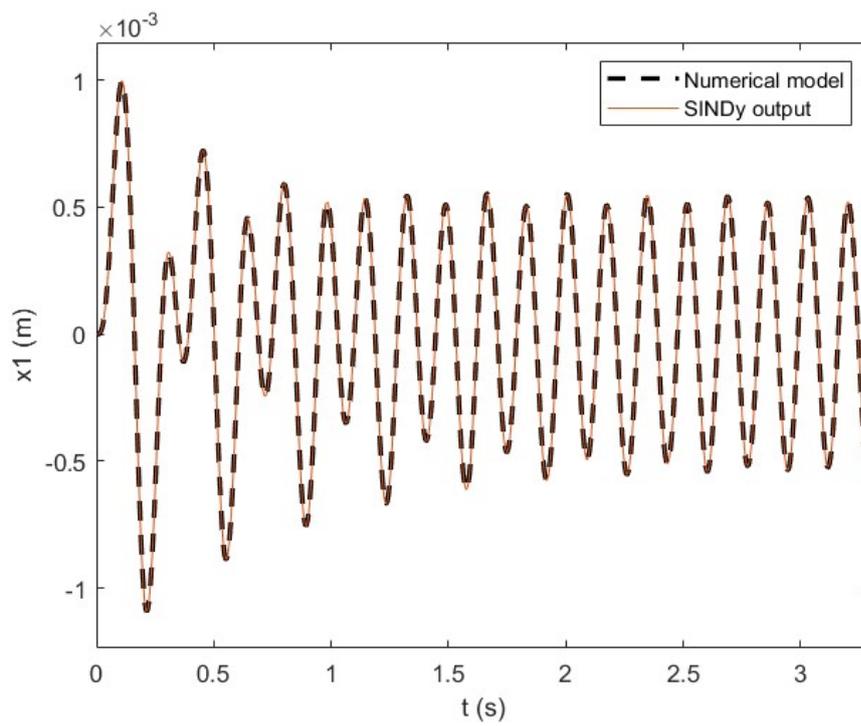


Figure 10: Comparison between PySINDy output and exact numerical output

3.4. Numerical friction model with noise

Noise is added to the state x_1 of model in section 3.3. The magnitude of the noise added is 0.00001 m. The segment length used for smooth finite difference is 31 units. The following are the testing input variables u (driving frequency 5.843 Hz), x_1 and corresponding PySINDy output:

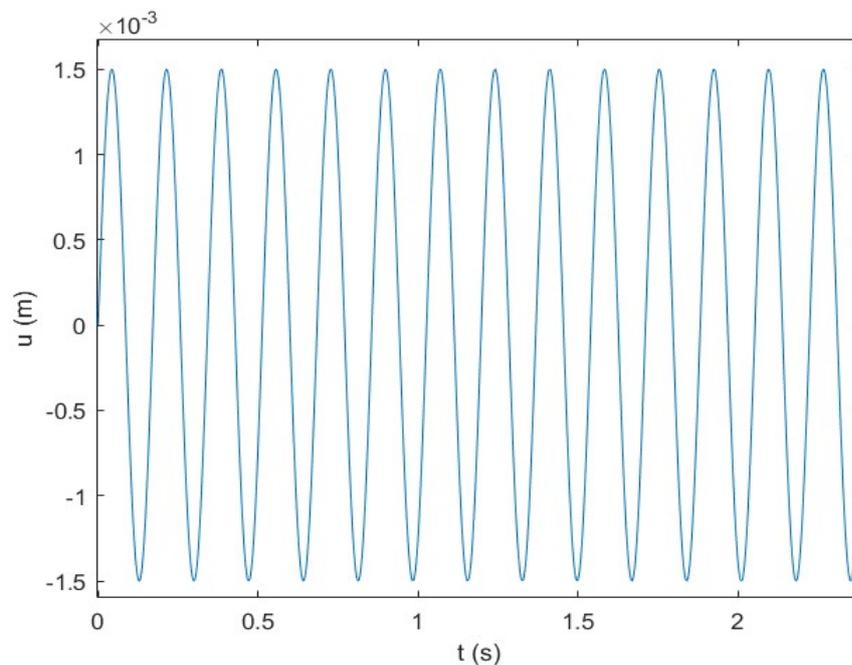


Figure 11: Base excitation

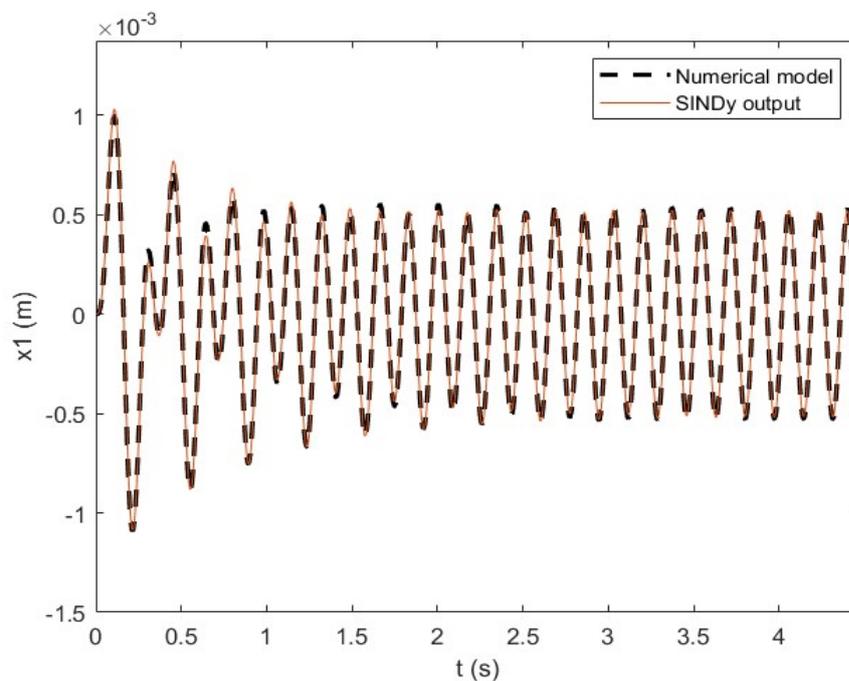


Figure 12: Comparison between PySINDy output and exact numerical output

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 0.999 \\ -356.888 & -0.233 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ -0.076Sgn(x_2) \end{bmatrix} + \begin{bmatrix} 0 \\ 353.568u + 0.213\dot{u} \end{bmatrix} + \begin{bmatrix} 0 \\ -0.004Sgn(x_1) + 0.001Sgn(u) \end{bmatrix} \quad \text{Equation 16}$$

It can be observed that there is a deviation in the coefficient compared to the actual numerical Equation 14. This deviation in coefficients is compensated by the additional terms as shown in above Equation 16

3.5. Experimental friction model

A single degree of freedom system is analysed in laboratory described in Figure 13. The base is excited using a rotor.

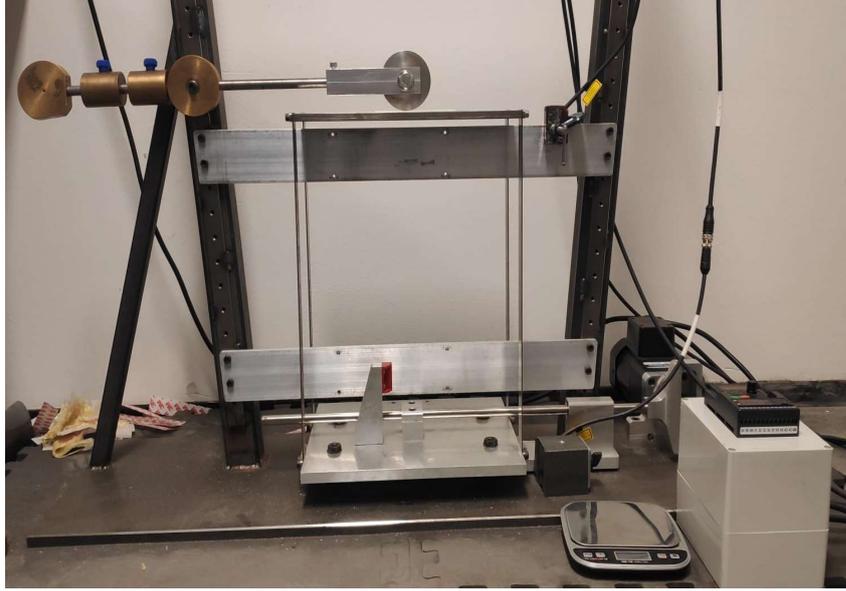


Figure 13: Experimental set up of SDOF system

The base excitation and displacement of mass is given as input to PySINDy in the same way as in section 3.3. the equation of motion of this system is given as follows:

$$m\ddot{x} + c\dot{x} + kx + F_f \cdot \text{Sgn}(\dot{x}) = ku + c\dot{u} \quad \text{Equation 17}$$

$$\therefore \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{c}{m} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ -\frac{F_f \text{Sgn}(x_2)}{m} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{ku + c\dot{u}}{m} \end{bmatrix} \quad \text{Equation 18}$$

For the system parameters such as friction coefficient and damping, a free vibration analysis is performed on the experimental setup and the values are calculated using linear decrement curve and exponential logarithmic decrement curve, respectively. After considering all the system parameters, the equation of motion can be written as:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -358.706 & -0.0658 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ -0.0856 \text{Sgn}(x_2) \end{bmatrix} + \begin{bmatrix} 0 \\ 358.706u + 0.0658\dot{u} \end{bmatrix} \quad \text{Equation 19}$$

The u with driving frequencies 2.013 Hz, 2.583 Hz, 4.611 Hz, 5.242 Hz and corresponding x_1 are taken as inputs for PySINDy. The output equation of motion of PySINDy is as follows:

$$\begin{aligned} & \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} \\ &= \begin{bmatrix} 0 & 1 \\ -359.896 & -0.176 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ -0.068 \operatorname{Sgn}(x_2) \end{bmatrix} \\ &+ \begin{bmatrix} 0 \\ 357.466u + 0.8\dot{u} - 0.068 \operatorname{Sgn}(x_1) - 0.068 \operatorname{Sgn}(x_2) + 0.023 \operatorname{Sgn}(u) - 0.034 \operatorname{Sgn}(\dot{u}) \end{bmatrix} \end{aligned}$$

The following are the testing input variables u (driving frequency 5.843 Hz), x_1 and corresponding PySINDy output:

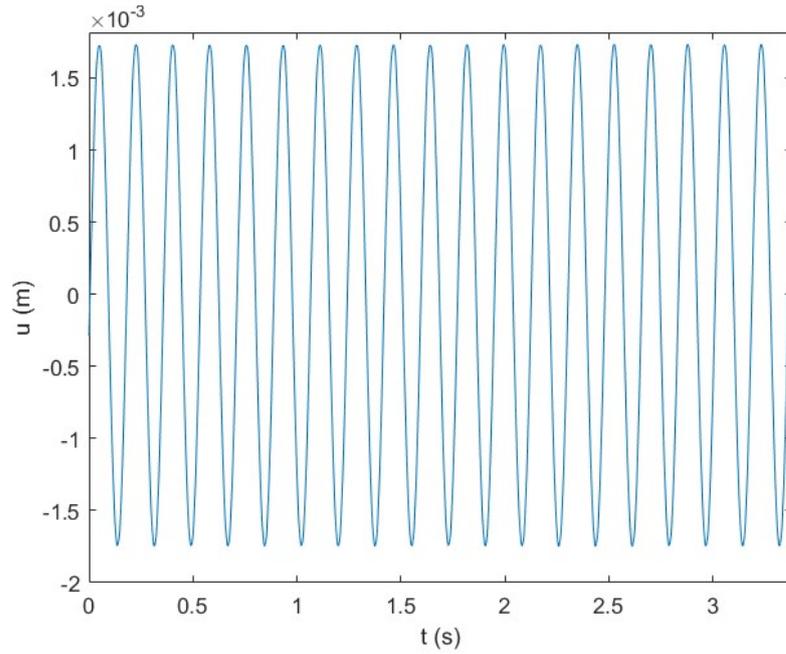


Figure 14: Base excitation

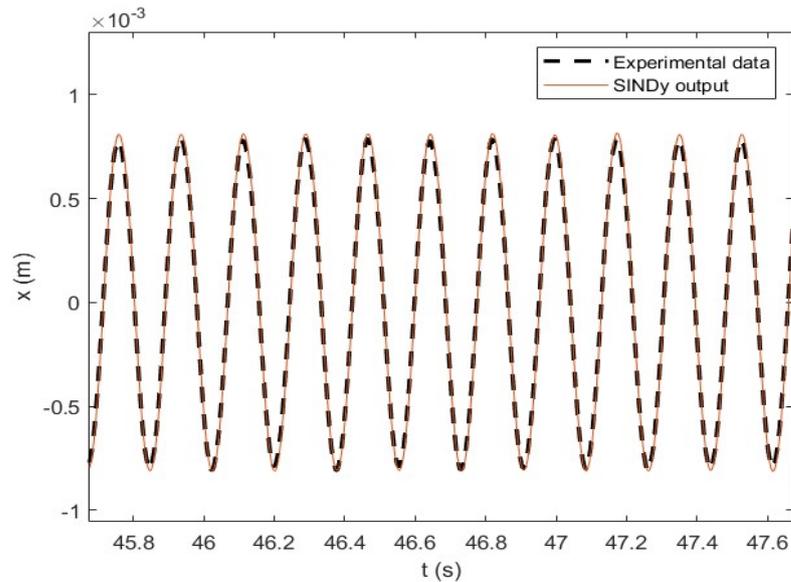


Figure 15: Comparison between PySINDy output and exact numerical output (only steady states are compared as we don't have the experimental data for transient response)

The deviation in equation is because the x_1 input had only steady state part and no transient part due to unavailability of data.

4. Quantification of error in equation of motion calculated by PySINDy

The absolute error at each time instant is defined \mathbf{E} as $\sqrt{(x_{1,actual} - x_{1,PySINDy})^2}$ while the total error is given by Root Mean Square Error (RMSE) as $\sqrt{\frac{\sum_{i=1}^{i=n}(x_{1,actual,i} - x_{1,PySINDy,i})^2}{n}}$. The Table 1 gives the RMSE of each dynamical system discussed in section 3. The plots of \mathbf{E} are shown in Appendix A for reference.

Table 1: Root Mean Square Error (RMSE) of different systems

Case	RMSE (m)
Duffing oscillator without noise	0.0024
Duffing oscillator with noise	0.0017
Numerical friction model without noise	1.42×10^{-6}
Numerical friction model with noise	5.01×10^{-6}
Experimental friction model	8.933×10^{-5}

Hence it can be observed that PySINDy gives fairly accurate solution with low RMSE-value.

5. Concluding remarks and future work

PySINDy works very well if the input data has no noise in it. The inclusion of noise can be dealt by changing the threshold of sparsity specified in the objective function of optimizer. It is also observed that the length of segments in smooth finite difference played a key role in simulation. This is because, the velocities are not given as the direct input to the PySINDy (in this case) rather velocities are calculated using numerical differentiation of displacement data. Hence if the displacement data has a considerable amount of noise and the segment length is small, then there would not be accurate smoothing of displacement data and hence the velocities calculated will have high fluctuations than ideal case.

It is not possible to get the equation of motion by blindly applying the SINDy algorithm to the data available. This is because the type of terms required in the output are specified by the user which indirectly means that the user should have some knowledge of physics associated with the experimental or numerical system. PySINDy gives equation of motion even when the terms specified in candidate function library are incorrect. This is due to the fact that SINDy algorithm tries to overfit the data using the specified incorrect terms and the solution fits the training data however fails for testing data. In case of very noisy measurements, velocimeters are recommended

The availability of transient response is important due to fact that there can be more combinations of terms which will fit the steady state response compared to the terms that will fit transient as well as steady state response. Also, more information about system parameters is hidden in the transient response while the steady state follows the forcing.

The transient response needs to be collected for experimental system and an effort should be made to use SINDy algorithm to identify the experimental system more accurately. The motion observed in the case of friction is continuous motion due to small magnitude of friction force. With increase in friction force, a stick slip phenomenon will take place which will be more complex in nature. The complexity will be due to the fact that in case of stick slip motion, there is point where the velocity is zero and the stiffness force is opposed by static friction force. This static friction force has a certain value which is not incorporated while calculating the continuous motion. Hence to include this sticking phenomenon, we will have to add additional constraints in PySINDy model by specifying some extra variable. This effect needs to be taken into account for future work. The effect of more contact points generating

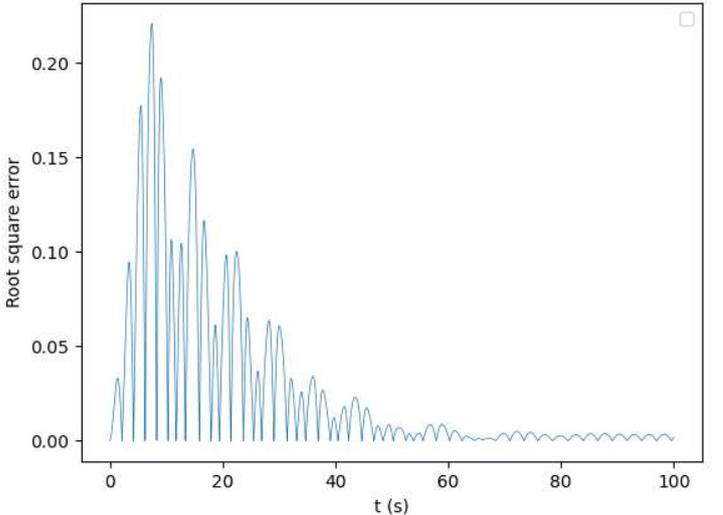
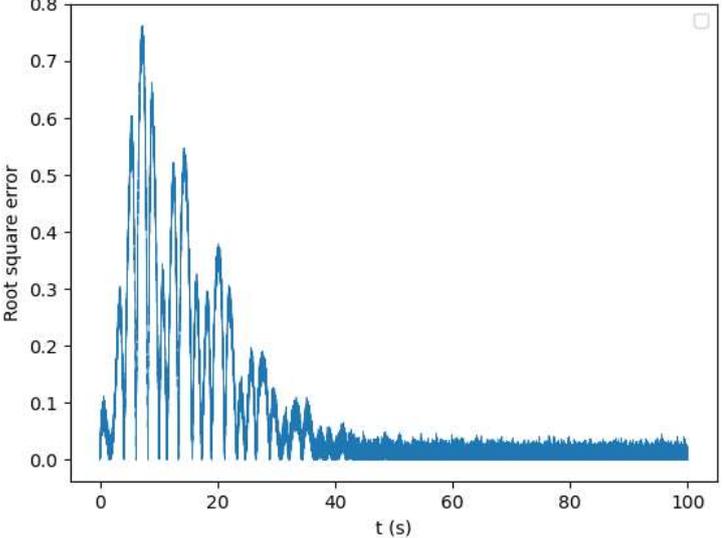
the friction force can be studied for future work. The suitability of SINDy algorithm for multi degree of freedom system needs to be checked.

6. Appendix

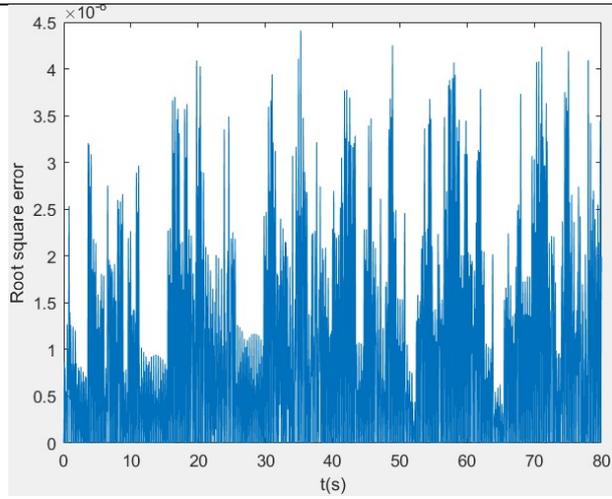
6.1. Appendix A

The graphs for E corresponding to the different systems are as follows:

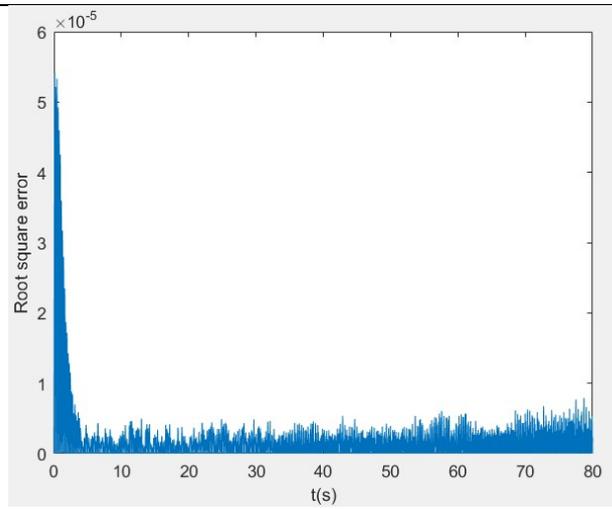
Table 2: Absolute error (m) in calculated PySINDy model

Case	$E = \sqrt{(x_{1,actual} - x_{1,PySINDy})^2}$
Duffing oscillator without noise	
Duffing oscillator with noise	

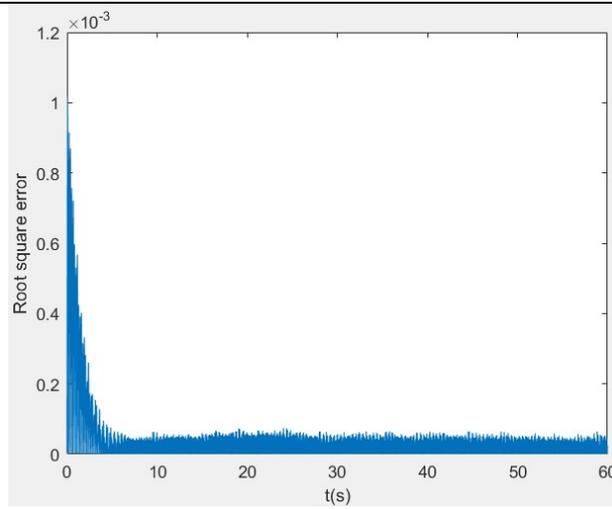
Numerical friction model
without noise



Numerical friction model with
noise



Experimental friction model



6.2. Appendix B

Initially an attempt was made to find the equation of motion in form of $\dot{X} = f(X)$ where \dot{X} is velocity of the single degree of freedom. However, the following derivation shows that we need to specify integral terms in PySINDy to get the correct solution. From state space form we get:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{c}{m} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ -\frac{F_f \text{Sgn}(\dot{x})}{m} = F_f \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{ku + c\dot{u}}{m} = F_g \end{bmatrix}$$

We can write $\dot{x} = \dot{x}_1 = x_2$ and

$$x_2 = \int_0^t -\frac{k}{m} x_1 dt - \int_0^t \frac{c}{m} x_2 dt + \text{some function of forcing}$$

$$x_2 = \int_0^t -\frac{k}{m} x_1 dt - \int_0^t \frac{c}{m} \dot{x}_1 dt + \text{some function of forcing}$$

$$x_2 = \dot{x} = \int_0^t -\frac{k}{m} x_1 dt - \frac{c}{m} x_1 + \text{some function of forcing}$$

As PySINDy cannot support integral terms in candidate function library, this approach was not adopted further.

6.3. Appendix C: Meeting overview

Following are the overviews of the meeting

Meeting on 13th October 2022

- The candidate library chosen needs to be changed and recommendation to have consultation with members of DVU group

Meeting on 25th October 2022

- Discussion of results using the approach specified in Appendix B
- The obtained result can not be compared with the state space form i.e., the expression of velocity obtained in Appendix B is different than what was obtained from PySINDy
- Recommendation to use PySINDy such that we get equations in state space form
- Try using PySINDy on duffing oscillator with and without noise, numerical model of friction with and without noise

7. Bibliography

- [1] L. Marino, “Dynamic analysis of mechanical systems with Coulomb friction,” University of Oxford, 2021.
- [2] S. L. Brunton, J. L. Proctor and J. N. Kutz, “Discovering governing equations from data by identification of nonlinear dynamical systems,” *PNAS*, 2016.
- [3] E. Kaiser, J. N. Kutz and S. L. Brunton, “Sparse identification of nonlinear dynamics for model predictive control in the low-data limit,” *Proceedings of The Royal Society*.
- [4] M. Didonna, M. Ciavarella, A. Papangelo, M. Stender, F. Fontanela and N. Hoffmann, “Reconstruction of Governing Equations from Vibration Measurements for Geometrically Nonlinear Systems,” *Lubricants*, 2019.
- [5] J. Fehr, A. Kargl and H. Eschmann, “Identification of Friction Models for MPC-based Control of PowerCube Serial Robot,” *arXiv*, 2022.
- [6] M. Lin, C. Cheng, Z. Peng, X. Dong, Y. Qu and G. Meng, *Nonlinear dynamical system identification using the sparse regression and separable least squares methods*, 2021.
- [7] M. B. de Silva, K. Champion, M. Quade, J.-C. Loiseau, J. N. Kutz and S. L. Brunton, “PySINDy: A Python package for sparse identification of nonlinear dynamical system from data,” *Journal of Open Source Software*, 2020.
- [8] Kaptanoglu, “PySINDy: A comprehensive Python package for robust sparse system identification,” *Journal of Open Source Software*, 2022.
- [9] Y. Ren, C. Adams and T. Melz, “Uncertainty Analysis and Experimental Validation of Identifying the Governing Equation of an Oscillator Using Sparse Regression,” *Applied sciences*, 2022.