Discovering governing equations from data: Sparse identification of nonlinear dynamical systems

Group 21
BITS F464 - Machine Learning

Ayushi Dubal 2019A7PS0099G
Naresh Chavan 2019B5A70638G
Shaurya Tandon 2020A4PS1556G
Reddygari Sai Saran Reddy 2019AAPS0191G

Acknowledgement

- We are grateful to authors Steven L. Brunton, Joshua L. Proctor , and J. Nathan Kutz for their inspiring work in the paper “Discovering governing equations from data by sparse identification of nonlinear dynamical systems”.
- Our sincerest gratitude to Professor Harikrishnan NB who guided us and was willing to lend an ear at all times.
- We would also like to thank the teaching assistants Tanmay Devale, Param Biyani, and Ramanathan Rajaraman for their steady dedication and help.
- We would like to thank the authors for publishing SiNDy as an open-access Python library.

- **Honour Code: We shall be honest in our efforts and will make our parents proud.**

Introduction

- Many physical systems around us have nonlinear equations that govern them i.e., a linear change in the inputs does not produce a linear change in the outputs.
- The equations defining these systems may be multivariate, and have higher order polynomial, logarithmic, and sinusoidal terms.
- Keeping in mind a very fundamental dogma: *most physical systems have only a few relevant terms that define the dynamics of it; hence the governing equations turn out to be sparse in a high-dimensional nonlinear function space.*

Problem Definition

- Our goal is to discover the governing equation of (possibly noisy) measurement data of nonlinear dynamical systems. Scientifically speaking, we have the experimental data for first derivatives over time along with the coordinates of the system. We then find the equations of the system using the data.

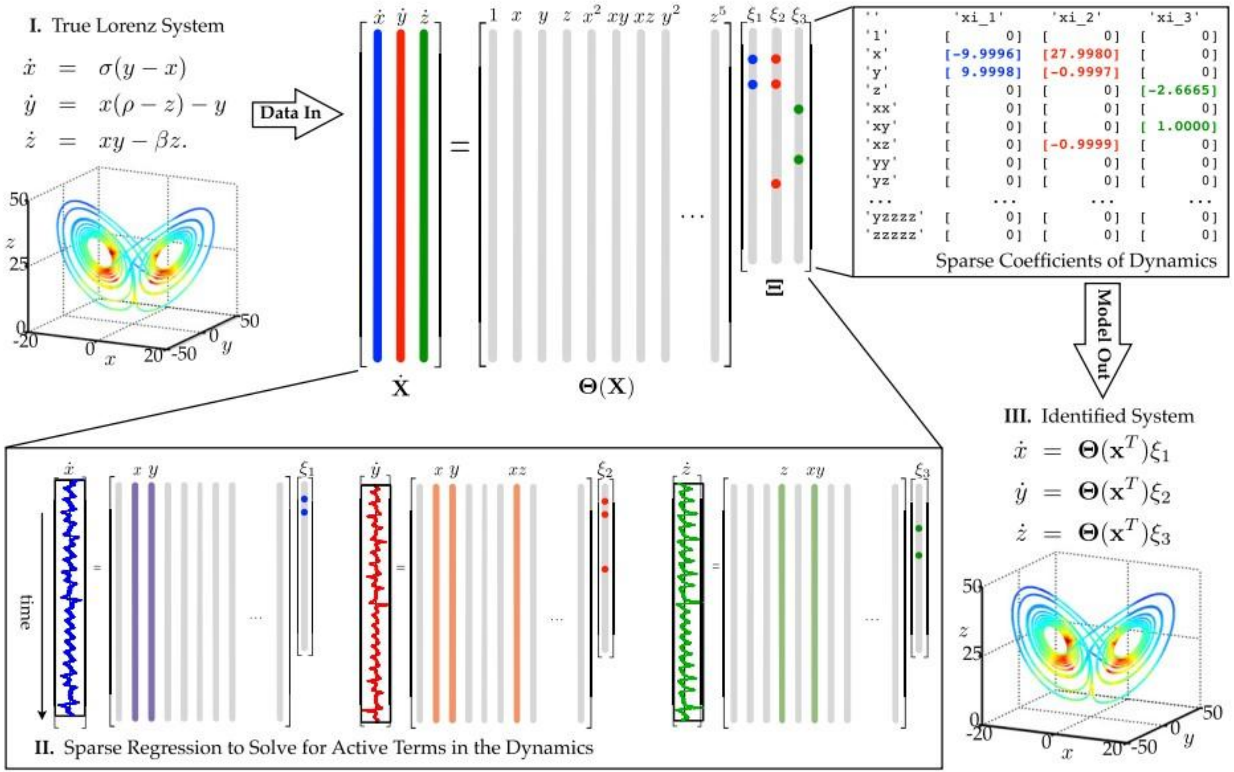
$$\frac{d}{dt}\mathbf{x}(t) = \mathbf{f}(\mathbf{x}(t))$$

Objectives

- To reproduce and extend the results of the paper by analysing the effect of noisy training data on the predicted systems.
- Finding the optimal size of data required to predict the nonlinear system correctly.
- Working with three different types of dynamical systems to test robustness of the algorithm
  - Linear 2D ODE (Damped Harmonic Oscillator)
  - Linear 3D ODE
  - Nonlinear ODE (Chaotic Lorenz System)

NOTE: Due to constraint on number of slides we chose to discuss about Chaotic Lorenz System

Methodology



$$\dot{\mathbf{X}} = \Theta(\mathbf{X})\Xi$$

Often, only ***X*** is available taking the derivative of ***X*** to get the LHS adds noise to the derivative. The authors suggest different ways to tackle this issue. One such way is by adding an ***ηZ*** term to the equation.

$$\dot{\mathbf{X}} = \Theta(\mathbf{X})\Xi + \eta\mathbf{Z}$$

Z is modelled as a matrix of ***i.i.d*** Gaussian entries with zero mean, and noise magnitude ***η***.

Methodology: How to get dataset

- We use the solve_ivp function from scipy.integrate package to generate the dataset.
- The solve_ivp method takes the following input:
  - A function that returns the first derivative value over time
  - Initial conditions for the system (x0_train)
  - Time series to calculate the coordinates over fixed interval of time

Experiment and Results

We have studied the dynamics for the Lorenz system. Following the methodology,

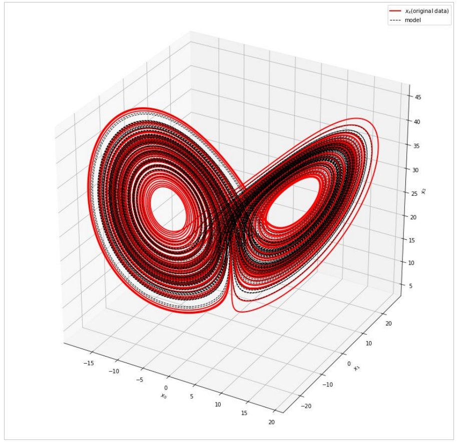
- Step-1: Generating data for the system of our choice using the known equations (Lorenz here).
- Step-2: Obtaining the best hyperparameters (optimal thresholds and feature libraries) using GridSearchCV (used time series split for cross validation).
- Step-3: Fitting the model with the best hyperparameters obtained from the previous steps.
- Step-4: Using the model, we simulate the system and compare the simulated data with the initial data used.

Experiments and Results

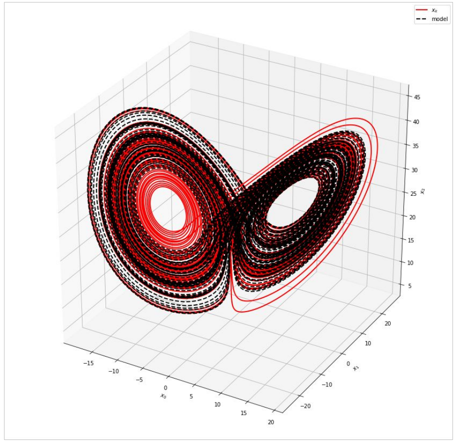
We study the effect of:

- The amount of noise in the data: As the level of noise increases, the system becomes more chaotic with new terms being introduced to the equation dynamics.
- The number of samples required at given levels of noise: For higher levels of noise we observe a higher required sample size for acceptable results.

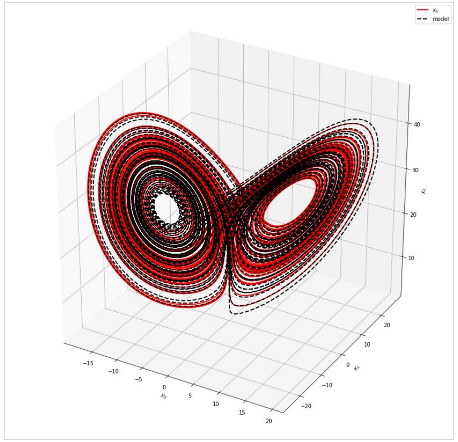
Results: Varying the level of noise



Noise level = 0

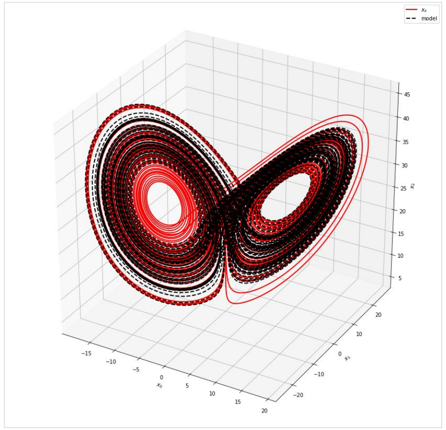


Noise level = 10-4

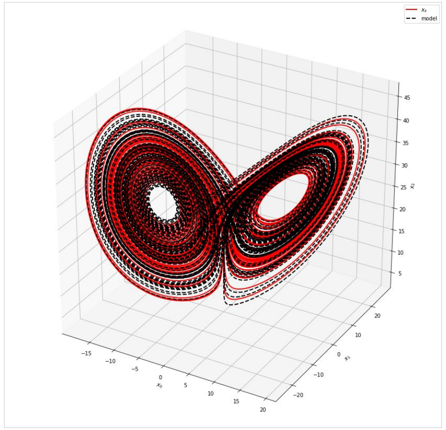


Noise level = 10-3

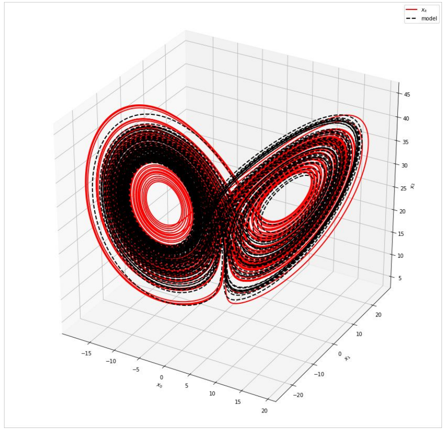
Results: Varying the level of noise



Noise level = 10-2



Noise level = 10-1

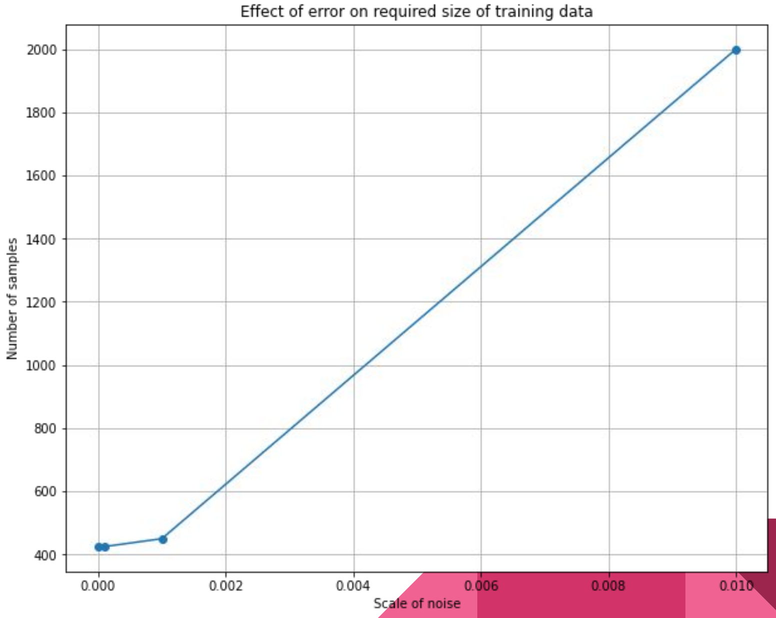


Noise level = 100

Discussion

Varying the noise in data:

- As we vary the scale of noise from e^{-4} to e^{0} we observe slight changes in the coefficients from e^{-4} to e^{-2} but starting e^{-1} we can observe more terms being introduced to the equations as the system becomes more chaotic and complex.
- Number of samples for given levels of noise increases with noise levels.



Conclusion

SINDy presents an approach that adds to prior work in symbolic regression by incorporating innovations related to sparse regression. There are numerous fields where this method may be applied: climate science, epidemiology, neuroscience, financial markets, chemical kinetic models in the human body, plasma dynamics, non-linear optic systems, complex mechanical systems.

Future Work

- The package does not support customizing the Lorenz coefficients - the inbuilt function assumes one set of coefficients

Challenges

- Choice of Coefficients: In a system where we have no prior knowledge of the system physics, we would have to proceed with a trial and error method to assess which coefficients should be observed.
- Library of Functions: Choosing the correct function basis is another challenge which can only be tackled using the trial and error method unless we have any prior knowledge on system physics.
- Controlling Noise: As is seen in the results, noise can significantly modify the dynamic equations of the system. In raw data obtained from tests/experiments, noise would hence be a primary concern.
- Amount of Data: For the lorenz system, we needed at least 500 samples of data entirely void of noise to observe acceptable results.
- Choice of Optimizer: We chose the STLSQ optimizer (which utilizes Ridge Regression) over any other optimizer in the SINDy library to avoid oversimplifying or results.

Code

Github:
https://github.com/naresh1205/Machine-Learning-BITS-F464-Term-Project.git

Nonlinear ODE (Chaotic Lorenz system):
https://colab.research.google.com/drive/1L3oa34uRSgNlOqPATIu1xwxZOsVEnQmC

Linear 2D ODE (Damped Harmonic Oscillator):
https://colab.research.google.com/drive/1C9LHa7Un-uD7qw8b0Xqg_QhnRtiNXply

Linear 3D ODE:
https://colab.research.google.com/drive/1lVz8oOETiWEinKMp_tA5N07[illegible]c53f-4yK

References

- Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz - https://www.pnas.org/doi/full/10.1073/pnas.1517384113
- Gustavo T. Naozuka, Heber L. Rocha, Renato S. Silva & Regina C. Almeida - https://link.springer.com/article/10.1007/s11071-022-07755-2