Robust Sparse Identification and Saddle Mediated Transport of Nonlinear Dynamics

Kadierdan Kaheman

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2022

Reading Committee:

Steven L. Brunton, Chair

J. Nathan Kutz

Sawyer Fuller

Mehran Mesbahi

Program Authorized to Offer Degree:
Mechanical Engineering

University of Washington

Abstract

Robust Sparse Identification and Saddle Mediated Transport of Nonlinear Dynamics

Kadierdan Kaheman

Chair of the Supervisory Committee:
James B. Morrison Endowed Career Development Professor Steven L. Brunton
Mechanical Engineering

The data-driven modeling approach has become increasingly popular to model the system where the first-principled approach is prohibitively challenging, such as finance, climate science, biology, epidemiology, etc. The Sparse Identification of Nonlinear Dynamics algorithm (SINDy) is a data-driven method that serves as an alternative to the first-principled modeling approach. The SINDy has the unique characteristic of identifying a sparse and interpretable model given the measurement data of the system of interest. However, the original SINDy can not identify implicit dynamics, limiting its ability to identify a dynamical system with rational functions. Moreover, common with many data-driven techniques, the SINDy is not robust to noise. This thesis proposes two important extensions that extend SINDy's ability to identify rational dynamics robustly (SINDy-PI) and improves SINDy's noise robustness by using state-of-the-art time-stepping constraints and automatic differentiation techniques (modified SINDy). We will also show SINDy's application in discrepancy modeling using various numerical examples and an experimental double pendulum. This discrepancy modeling framework of SINDy is advantageous when an imperfect model is already known. The work on the SINDy algorithm naturally brings out another important topic: how to design an experimental benchmark system for chaos, learning, and control. As a well-known classic textbook example, the multi-link pendulum on the cart system has been well studied. However, extensive documentation on such a

system's design, construction, and operation is missing from the literature. Thus, this thesis also provides a detailed tutorial on building a multi-link pendulum experimental system that is highly flexible, enabling a wide range of benchmark problems in dynamical systems modeling, system identification and learning, and control. The building of the pendulum hardware system is also for the future experimental validation of the saddle-mediated transport of the double pendulum. The transport of material in chaotic dynamical systems has been shown to be mediated by the presence of saddle points in the phase space. This perspective has been used to explain efficient bio-locomotion, develop energy-efficient transport routes in the solar system for space mission design, and understand chemical reaction kinetics. In particular, the mechanical double pendulum has three fixed saddle points and exhibits rich, chaotic motion. This thesis then provides a detailed analysis to establish a correspondence between the double pendulum and the planar circular restricted three-body problem. Next, we demonstrate numerically the existence of a family of unstable periodic orbits (UPOs) that organize the phase space trajectories. This involves a thorough characterization of the homoclinic and heteroclinic connections. These UPOs and connections establish a skeleton that enables high-level control of the itinerary or topology of a future trajectory in symbolic dynamics. Those numerical analyses on the double pendulum then serve as foundational work for the future experimental verification of the saddle-mediated transport of the double pendulum using the experimental system we built.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

GLOSSARY

ERA: Eigen-realization algorithm.

POD: Proper orthogonal decomposition.

DMD: Dynamic mode dycomposition.

NN: Neural network.

GR: Gaussian regression.

GA: Genetic algorithm.

GP: Genetic programing.

SR: Symbolic regression.

GE: Grammatical evolution.

AP: Analytical programming.

NARMAX: Nonlinear auto-regressive algorithms.

ADAM: Adaptive moment estimation.

ROM: Reduced-order modeling.

VAC: Variational approach of conformation dynamics.

SINDY: Sparse identification of nonlinear dynamics algorithm.

ODE: Ordinary differential equations.

STLSQ: Sequential thresholded least squares.

NP hard: Nondeterministic polynomial time hard.

LASSO: Least absolute shrinkage and selection operator.

WBPDN: Weighted basis pursuit denoising algorithm.

Enet: Elastic net.

STRidge: Sequentially thresholded ridge regression.

SSR: Sparse stepwise regression.

HRidge: Hierarchical ridge regression.

OMP: Orthogonal matching pursuit.

USDL: Unified sparse dynamics learning.

GLASSO: Group LASSO.

SGTR: Sequentially grouped threshold ridge regression.

SR3: Sparse relaxed regularized regression.

NPSLS: Non-convex penalty least-squares algorithm.

AIC: Akaike information criteria.

BIC: Bayes information criteria.

CV: Cross validation.

SubTSBR: Subsampling-based threshold sparse Bayesian regression.

E-SINDY: Ensemble-SINDy.

WSINDY: Weak SINDy.

PDE: Partial differential equation.

FFT: Fast Fourier transform.

FoMs: Figures of merits.

HSLDS: High-static-low-dynamic stiffness.

RK4-SINDY: Fourth-order Runge-Kutta SINDy.

nSINDY: Neural SINDy.

SISC: Subsampling and co-teaching.

LM: Levenberg–Marquardt algorithm.

SINY-SA: Sensitivity analysis SINDy.

EE: Elementary effects method.

SSE: Sum of squared errors.

CG: Conditional gradient.

SINDYFE: Feature engineering SINDy.

rr-SINDY: Regularized horseshoe SINDy.

ss-SINDY: Spike-and-slab SINDy.

UQ-SINDY: Uncertainty quantification SINDy.

MCMC: Markov Chain Monte Carlo.

CNN-SINDY: Convolutional neural network SINDy.

RF: Radio frequency.

PA: Power amplifier.

SINO: Sparse identification for nonlinear optical.

NMPC: Nonlinear model predictive control.

MPC: Model predictive control.

DoF: Degrees of freedom.

BVP: Boundary value problems.

SITE: Sparse identification of truncation errors.

CNN-AE: Convolutional neural network-based autoencoder.

MHD: Magnetohydrodynamic.

DNN: Deep neural network.

OASIS: Operable adaptive sparse identification of systems.

IPF: Idiopathic pulmonary fibrosis.

SINDY-LM: Levenberg-Marquardt based SINDy.

BZ reaction: Belousov-Zhabotinsky reaction.

SINDY-PI: Parallel and implicit SINDy.

TVRegDiff: Total-variation regularization difference.

VTOL: Vertical take-off and landing.

CPR: Counts per revolution.

PPR: Pulses per revolution.

IMU: Inertial measurement unit.

CNC: Computer numerical control.

UPOs: Untsable periodic orbits.

PCR3BP: Planar circular restricted three body problem.

ACKNOWLEDGMENTS

I want to thank Professor Steven L. Brunton and Professor J. Nathan Kutz for their friendship and mentorship during my Ph.D. Professor Brunton has been a tremendous support during my research and study. He is patient, supportive, and always full of fantastic research ideas. Moreover, he still amazes me with an easygoing personality, humor, and sharpness and will always be a role model for me. Professor Kutz co-advised me during my Ph.D. and is an outstanding researcher, advisor, and friend. He can provide excellent guidance on the research topic and is extremely good at teaching and explaining the research idea to the students. His confidence and great personality make it a pleasure to do research with him. I have learned a lot from them and still have a lot to learn. Thus, I sincerely thank their guidance and support during my Ph.D., without which this thesis would be impossible, and I am extremely grateful for having great advisors like them. I also want to thank my reading committee members, Professor Sawyer Fuller and Professor Mehran Mesbahi, for spending valuable time and providing helpful feedback on this thesis.

Additionally, I also want to thank all my coauthors, who have always been great resources and friends to me. I want to thank Benjamin Strom for his assistance during the early stage of my pendulum research and provided helpful insight into the pendulum project. I want to thank Eurika Kaiser for her excellent support during the early phase of my Ph.D., providing knowledge on control theory and dynamical systems. It's a great pleasure that we coauthored my first paper on discrepancy modeling, and I learned a lot from them. I want to thank Jason Bramburger, who has shown tremendous support for the saddle-mediated transport of the double pendulum project. As an expert in the dynamical

system, he is always ready to discuss the project and has also been a great resource and friend for me to learn more about the dynamical system. I also want to thank Urban Fasel for his support and valuable discussion on documenting the pendulum hardware. I also thank his help in making the videos for the pendulum hardware setup. I would also like to thank Daniel A. Messenger for his help during the modified SINDy project, Professor Shane Ross for his valuable feedback on the saddle-mediated transport project, Eamon McQuaide, and Veasna Thon for their support during the pendulum hardware project. Moreover, I would like to thank all my previous and current lab mates, including but not limited to Ariana Mendible, Jared Callaham, Aditya G. Nair, Benjamin Herrmann, Samuel H. Rudy, Isabel Scherl, Henning Lange, Joseph Bakarji, Steven Rodriguez, Krithika Manohar, Alan Ali Kaptanoglu, Zhe Bai, Abhay Gupta, Brian DeSilva, Tony Piaskowy, Zachary G. Nicolaou, and Megan Ebers. They have been great friends for me, and I value all the support they provided and all the interesting research ideas we discussed before. I would also like to thank Professor Ashis Banerjee, Professor Ashley Emery, and Professor Behçet Açikmeşe for their excellent lectures, which are fundamental for my research. I also want to thank Jay Gorasia and all my teammates at Tesla. I shared a fantastic summer with them during my internship, and they gave me a new perspective on implementing my research work into industrial applications. I also want to thank Daksh Dhingra and Jacob Beers for their friendship and the valuable research discussions.

Moreover, I would like to thank my earliest mentor Mr. Wang Yujiang, my primary school science teacher, who helped me reinforce my interest in science when I was a kid. Back in the days when the internet is not readily available, he has been the primary source of knowledge for me. I would also like to thank Professor Memtimin Gheni, for encouraging me in pursuing knowledge and higher education.

Finally, I would like to thank all my family members, especially my parents, for their help and support. They have always been highly supportive and patient. Moreover, they

give me helpful advice and perspectives whenever I needed.

Funding: This thesis is supported by part or by whole through the Army Research Office (ARO W911NF-17-1-0306, ARO W911NF-19-1-0045), the Air Force Office of Scientific Research (AFOSR FA9550-18-1-0200), the Air Force Office of Scientific Research (AFOSR FA9550-17-1-0329), Defense Advanced Research Projects Agency (DARPA- PA-18-01-FP-125), Washington Research Foundation Fund for Innovation in Data-Intensive Discovery, and National Science Foundation AI Institute in Dynamic Systems (grant number 2112085).

DEDICATION

To my hometown and its people.

Chapter 1
INTRODUCTION

Whether it is a heavenly body or things as abstract as the stock market, nearly everything around us that evolves in time and space thus result in *dynamical system*. Dynamical system can be modeled by some laws that governs them. Thus, a model has an explanatory ability to characterize the dynamical system's behavior making it useful for analytical analysis, state estimation, and control, etc. Those extraordinary abilities are why modeling is vital in robotics, astronomy, climate science, finance, epidemiology, biology, etc. However, obtaining an accurate model can be a challenging task. There are two dominant approaches for deriving a dynamical model: 1) deriving equations of motion from first principles governing physics and 2) data-driven modeling. For centuries, the first approach has been the dominant way to derive a model. However, depending on the system of interest, it can be a time-consuming task where lots of expert knowledge is needed. In the worst case, it can be prohibitively challenging, especially when the system is high-dimensional, nonlinear, and multi-scale, such as modeling biological network reaction, the human brain, weather, etc. With the ever-increasing amount of data and computational power, the data-driven system identification approach starts to play an important role where the traditional first-principled approach is prohibited. This modeling is possible since data contains information about the system dynamics, which is why many early triumphs in science benefited from the use of data.

For centuries, the scientist uses the data, whether gathered from observation or experiments, to prove their scientific theory and theoretical model. Inferring patterns and laws from the data play a vital rule from Galileo Galilei's (1564-1642) observation that confirmed the heliocentric model of Solar System [1] to Johannes Kepler's (1571-1630) discovery

of laws of planetary motion [2]. Sir Isaac Newton(1643-1727) then brings out Newton's laws of motion based on Kepler's work [3, 4]. Similarly, Gregor Mendel's (1822-1884) experiments on pea plants discovered laws of inheritance [5]. All those early triumphs in science benefit from the use of data. Nevertheless, albeit the ever-increasing amount of data available to us, unveiling the useful knowledge behind them can still be challenging. This fact might differ from the now naive view of philosopher Francis Bacon (1561-1626) who believed that any regularities in the data will be obvious to the observer once we have enough of them [6]. Although the early discovered physical laws have simpler forms, the physical laws we fascinated nowadays are much more complicated such as the Lagrangian and Hamiltonian of the mechanical system [7–9] or partial differential equations of fluid filed [10] [11, 12]. Thus, efforts need to be made to extract useful information from the data. Moreover, the unavoidable measurement noise will make discovery of useful information challenging [12–15]. Thus, a robust, computationally efficient algorithm that could extract useful information from the data without any prior knowledge is in desperate need of modern-day researchers.

1.1 Early Approaches on Inferring Equations and Laws from Data

In the 1960s, the scientist hopes to design an expert system that could explain the observed data. Numerous programs were developed during that time, including DENDRAL, which could use experimental data with the chemistry knowledge to produce possible chemical structures that explain the data [16]. DENDRAL and other programs derived from it relies on the expert knowledge to interpret the data and are far away from discovering the underlying physical law of the data. However, it reflects the *knowledge principle* of AI experts at that time, that *knowledge is power*, and it is an early trial of the experts to know more hidden information from the data [17, 18].

In the late 1970s, Langley (1977) introduced a program named BACON, wishing to achieve scientific-like behavior similar to the early programs DENDRAL [19]. In its early stage, the BACON could only discover the empirical laws, yet it might be the first program

that could perform equation discovery from the data. In 1979, the invention of BACON3 allowed the detection of the simple physical law, including ideal gas law and Kepler's third law, etc [20]. The success of the BACON3 comes from its ability to collapse the hypotheses on top of each other [21]. Two years later, the BACON5 was introduced, which allows the symmetric assumptions to be made for the physical law discovery [6].

Ever since the introduction of BACON, many other algorithms were developed to identify numeric laws [22]. It is noteworthy though that the BACON uses rules, like the constancy and trend between the data to discover the laws. Those rules have to be predefined by the human expert. The use of expert knowledge contradicts the idea of "solving problems without being explicitly programmed" by Arthur Samuel. He claims "programming computers to learn from experience should eventually eliminate the need for much of this detailed programming effort" [23]. So how could we achieve this? A *natural selection process* is one way to go.

In 1975, John H. Holland published a book that makes him become the father of the Genetic Algorithm. The Genetic Algorithm(GA) is an algorithm that performs the optimization task using bio-inspired operators such as selection, crossover, and mutation [24]. The first few years after the publication, the book did not gain much attention due to the computational limits of that time. Nevertheless, it increased its popularity after people realize that GA could tackle the fields where the standard approach does not apply [25]. At its early stage, the GA still can not discover the equation of motion or the underlying laws of the given data. In 1986, D.S. Broomhead et al. published their result on extracting the qualitative dynamics from the experimental data. In this work, they reconstructed the system behavior using the embedding method based on Taken's theorem [26, 27]. One year later, James P. Crutchfiel et al. published a work that could discover the equation of motion from the time series data [28]. In this work, the author tries to infer the equation of motion by time-series data using several nonlinear bases. The minimum embedding dimension is determined by the trade-off between the model entropy and the expansion order of the nonlinear basis. The final model is selected using *law of parsimony* (Occam's

Razor) [28]. This work could have several improvements — one of them being generate the nonlinear basis dynamically. In 1985, Cramer brought up the idea of the first tree-based Genetic Programming (GP) [29]. Later in 1992, John R. Koza [30, 31] implemented the GP for varies applications, including the GP-based Symbolic Regression (SR). Both the GP-based SR and Crutchfiel et al.'s approach tries to fit the nonlinear basis to a given time series data. Also, they all need to perform model selection process. The difference though is that the GP-based SR has a much larger search space. With the intelligent combination of nonlinear basis fitting and model selection through fitness function, GP-based SR becomes a powerful tool to infer hidden information behind the data.

1.2 Genetic Programming based Symbolic Regression

Genetic Programming has been used in many applications [31, 32], and this evolutionary computation method mainly mimics the natural process happens in nature: mutation, crossover, selection [30, 33]. Note that the selection plays an important role in the GP. Common approaches for selection are *fitness-proportional selection*, *tournament selection*, *elitism*, etc [34] [35]. The GP consists of the following: A terminal set which is set of inputs or constant; A function set that constructs the potential solution; A fitness function that evaluates the fitness of each individual; Several parameters that determine the mutation or crossover rate and finally a termination criteria [30] [31]. To get the desired solution, one must make sure that the GP satisfies the sufficiency requirement. This requirement guarantees the correct structure one wish to find out lives in the possible space formed by the terminal set and function set. Moreover, one has to make sure that the function set takes in all values it might receive gracefully, which is called the closure property [34]. Before getting the answer one desired, the first step one has to do is the initialization of the GP, approaches like *grow* and *full* could be used [34]. The key to the initialization step is the generation of diversified individuals. Thus the method like *ramped-half-and-half* is been introduced [36]. A modification of the GP results the Grammatical Evolution (GE) developed by Ryan et al. (1998) [37]. The difference between the GE and GP is that the

former could be generalized into any programming language [31]. This natural selection process, when used to perform model discovery, could be quite powerful and will result in GP (evolutionary) based Symbolic Regression. Unlike the classical regression method, which tries to determine the parameters of a given model [13], the task of the Symbolic Regression includes both finding a correct structure of the model and its corresponding parameters [30]. The previous SR method that based on the GP or GE uses an evolutionary approach to determine the parameters of the system model. This approach will increase the computational time needed to find out the correct parameters. Analytical Programming (AP) could overcome this problem by using the nonlinear fit to lock down the values of parameters [31]. The AP is still based on the GA idea, the difference is it uses the integer index to represent the individual [31, 38].

Though the GP-based SR is quite powerful and has been implemented successfully for many applications, it still has drawbacks. One of them is the *bloat* [30]. One easy fix to this problem is setting the threshold for how deep a code could be [30]. Other more sophisticated solutions include *pseudo-hillclimbing* to restrict the tree growth [39], using linear [40] or nonlinear weight [41] to combine the effect of code size when calculating the fitness. To balance the relative significance of prediction error and individual complexity Zhang et al. (1995) formulated the approach that could estimate the tree size growth [42]. To avoid using user define parameters to balance the fitness and size, approaches like *lexicographic parsimony pressure* [43] or *multi-objective methods* [44] could be used. Bongard et al. (2007) [13] mentioned another parameter-free way to overcome the tree size growth problem by *snipping* method. They randomly select part of the tree and replace it by the constant that best represent the output of the selected region. Moreover, to choose the best model, Bongard et al. used a process called *automated probing* that could disambiguate the competing model. Along with the *partitioning* process, they could treat each variable separately so that the search space grows polynomially. With all those improvements, they are the first to generate the symbolic equation for nonlinear coupled system automatically [13]. However, the search space of the high dimensional system is still substantial. To overcome

this, one could use the parallel computing method to increase the search speed [45, 46]. Later in 2008, Schmidt et al. come up with the fitness predictor that could reduce the computational cost further [47]. With all those innovations and improvements, Schmidt et al. (2009) come up with an evolutionary computing based symbolic regression method that could extract the free form conserved law from the measurement data [14]. Although GP has been brought up for many years, many open problems need to be solved [48]. Thus other approaches have been made after the publication of [14] to find out governing physical law. For example, [49] tries to search for the simplest Lagrangian that describes the system's behavior. Other references and more recent reviews of the GA can be seen at [50–55]

1.3 Other Data-Driven Modeling Approaches

There are many other data-driven approaches to identify a model besides GP-based SR mentioned in Sec. 1.2, and it is a pretty large and active topic. It is impossible to provide a detailed overview of all data-driven modeling approaches in just a few pages. Thus, this section only tries to overview some of the most commonly seen techniques in data-driven modeling. Based on the system those data-driven modeling algorithms are trying to model, they can be categorized as linear, nonlinear, hybrid, parametric, non-parametric, etc [56]. Some of the well-known methods are the linear method, eigen-realization algorithm (ERA), proper orthogonal decomposition (POD), dynamic mode dycomposition (DMD), operator inference [57, 58], nonlinear auto-regressive algorithms, Gaussian process, neural network (NN), Koopman theory, to name only a few [11, 59].

Among all those different approaches, the study of linear system identification is mature. The linear method tries to identify the model of the linear system through time or frequency domain [60, 61]. The time-domain model is frequently used in the controls application, while the frequency domain model is common in vibration and modal analysis. In the time domain, the identification can be split into continuous-time and discrete-time modeling [62, 63]. Some good references on linear system identification can be seen

in [64–67]. However, the linear approach can not tackle the ubiquitous nonlinear problem well. When dealing with nonlinear system, the methods like nonlinear auto-regressive algorithms (NARMAX) [68, 69], Gaussian process [70–75], NN [76–84], Koopman theory [85–88] can be used to model the nonlinear effect. The NARMAX approach tries to identify the model from the system input to system output given a time-delayed state and input measurement [89]. It can provide us with an equation that describes the relationship between the system input and output. It has a much more intuitive representation of the nonlinear dynamics than the Gaussian process and NN approach. NARMAX has been successfully applied to various applications [90–97].

The Gaussian process is an alternative method to model the nonlinear system, and it is a useful tool when the variance of prediction error is needed [98]. However, the Gaussian process is computationally heavy, and an approximation approach has been made to make it more suitable for big data applications [99, 100]. The Gaussian process modeling has been successfully applied to controls application [101–104], modeling wind turbine power curve modeling [105], human motion [106], large-scale terrain [107], astronomical time series [108], etc.

NN is another popular approach widely used nowadays for modeling nonlinear systems. NN models the input and output relationship by adjusting the weights and biases of a network, and it is a black-box modeling approach. Usually, this adjustment of weights and biases is made by using optimization approaches such as adaptive moment estimation (ADAM). NN has a wide application [109–122].

Recently, the Koopman modeling framework also got lots of attention. The Koopman theory framework tries to identify a coordinate transformation that transfers the nonlinear system into a linear one [123–125]. This is an alternative way to model the nonlinear system and a very tempting one. By getting a linear representation of the original nonlinear system, we can use the well-developed linear system theory to analyze the behavior of the embedded linear system and use it for control and estimation. However, finding this transformation of the coordinate is not a trivial thing to do. One well-known way to find

such an embedding is by using DMD [126–128]. However, the DMD does not perform well for a highly nonlinear system since it is based on linear measurement [87, 126, 129]. To overcome this issue, the Extended DMD [87, 130], kernel DMD [131, 132] and variational approach of conformation dynamics (VAC) [133, 134] are proposed. The Extended DMD and kernel DMD build a library of nonlinear measurements to find a linear operator that approximates the nonlinear dynamics. But those two methods are not guaranteed to work and might face closure issues. The NN can be used to approximate the Koopman operator to overcome those issues. Due to the NN's ability of approximating the arbitrary nonlinear function, it can be used to automatically select the Koopman observables, which facilitates the data-driven coordinate transformation [135–141]. The Koopman theory has been successfully applied to identification [142, 143], estimation [144, 145] and control of nonlinear system [146–153]. We recommend [86, 154–157] for reviews on the Koopman theory.

Reduced-order modeling (ROM) aims to find out a low dimensional representation of the original high dimensional system of interest. One way to get reduced order model is by collecting experimental or simulation data from the high dimensional system and use data-driven approach such as NARMAX, ERA [158–161], NN, DMD [127, 128, 162–168] to identify a low dimensional approximation. Another way to get ROM is to project the high dimensional system into low dimensional subspace using methods like POD [169–176].

1.4 Contribution

This thesis has the following contributions. It provides a detailed literature review on SINDy with many recent developments and variants. Next, SINDy-PI, a robust algorithm that identifies implicit or rational dynamics from measurement data, is introduced. Then, we present modified SINDy, which improves the noise robustness of the SINDy framework and can correctly identify the sparse nonlinear dynamics in the high noise regime. We also show the SINDy algorithm is a valuable tool in the discrepancy modeling framework to learn the missing dynamics that explain the difference between the user model and

measurement data. Our work on data-driven modeling motivated us to develop a multi-link pendulum on the cart benchmark system for studying chaos, learning, and control. We then show a detailed tutorial on building a multi-link pendulum on the cart system. We open-source our design and provide freely available data sets collected using the experimental setup. Those data sets are valuable resources for the machine learning community to test different data-driven modeling algorithms. Finally, we introduce the numerical analysis of the saddle-mediated transport of the double pendulum. We show that the double pendulum is an analog system of the planer restricted three-body problem. Then we show it has rich and exciting dynamics such as homoclinic and heteroclinic orbits. Those orbits help the saddle transport the double pendulum. This work serves as a foundational work for future experimental validation of the saddle-mediated transport of the double pendulum.

1.5 Outline

This thesis is organized as follow: In Chap. 2, we introduce necessary background information of the SINDy algorithm. In Chap. 3 we introduce SINDy-PI: a robust algorithm for parallel implicit sparse identification of nonlinear dynamics. In Chap. 4, we introduce modified SINDy, which simultaneously identifies nonlinear dynamics and extract noise probability distributions from data. In Chap. 5, we show SINDy can be used in the discrepancy modeling framework and identify missing dynamics between measurement data and user model. In Chap. 6, we provide a detailed documentation on how to build a multi-link pendulum on the cart system. In Chap. 7, we perform numerical analysis on the double pendulum and show in theory it is possible to achieve saddle mediated transport of double pendulum. Finally, in Chap.8, we summarize our work.

Bibliographic Notes: This thesis is based upon the following publications:

- The Chap. 3 uses the work by Kaheman et al. on SINDy-PI [177].
- The Chap. 4 uses the work by Kaheamn et al. on modifed SINDy [15].

- The Chap. 5 uses the work by Kaheman et al. on discrepancy modeling using SINDy [178].

- The Chap. 6 uses the work by Kaheman et al. on building the pendulum hardware [179].

Chapter 2

BACKGROUND: SPARSE IDENTIFICATION OF NONLINEAR DYNAMICS ALGORITHM

2.1 Original SINDy Algorithm

In Sec.1.2 we mentioned how GP-based SR is a powerful tool to infer patterns from data. However, GP base SR does have limitations such as high computational cost and large search spaces. On the other hand, NN and Gaussian process can model the input-output relationship of the data well but lacks physical interpretability. This makes the sparse identification of nonlinear dynamics algorithm (SINDy) a favorable choice when a sparse and interpretable solution is needed.

In the sparse identification of nonlinear dynamics framework [180], we are usually interested in discovering the underlying dynamics of the first-order ordinary differential equations (ODE):

$$\frac{d}{dt}\mathbf{x}(t) = \boldsymbol{f}(\mathbf{x}(t)), \tag{2.1}$$

with time-dependent state vector $\mathbf{x}(t) = [x_1(t), \ldots, x_n(t)] \in \mathbb{R}^{1\times n}$. One assumption that we use is that for many system of interest, the nonlinear equation $\boldsymbol{f}(\mathbf{x}(t))$ usually contains few terms such that it is sparse in the space of the candidate functions. To identify the nonlinear dynamics using SINDy, one must perform following steps.

Step 1: Sample the system state from time $t_1, \ldots, t_m$ to form the time-history data matrix $\mathbf{X} \in \mathbb{R}^{m\times n}$, $\dot{\mathbf{X}} \in \mathbb{R}^{m\times n}$ where m is the data stream length and n is the number of

system states:

$$\mathbf{X} = [\mathbf{x}(t_1); \mathbf{x}(t_2); \cdots ; \mathbf{x}(t_m)], \tag{2.2a}$$

$$\dot{\mathbf{X}} = [\dot{\mathbf{x}}(t_1); \dot{\mathbf{x}}(t_2); \cdots ; \dot{\mathbf{x}}(t_m)]. \tag{2.2b}$$

Step 2: Define the candidate functions $\theta(\mathbf{X})$ and stack the candidate data stream into a matrix $\boldsymbol{\Theta}(\mathbf{X})$:

$$\boldsymbol{\Theta}(\mathbf{X}) = [\theta_1(\mathbf{X}), \theta_2(\mathbf{X}), \theta_3(\mathbf{X}), \cdots , \theta_v(\mathbf{X})], \tag{2.3}$$

where $\boldsymbol{\Theta}(\mathbf{X}) \in \mathbb{R}^{m \times v}$. The candidate functions can be anything that may exist in the system's actual dynamics $\boldsymbol{f}(\boldsymbol{x})$. It could be trigonometric or polynomial function such as $\theta_i(\mathbf{X}) = \cos(\mathbf{X})$, $\theta_j(\mathbf{X}) = \mathbf{X}$, etc.

Step 3: Perform the sparse regression

$$\arg\min_{\boldsymbol{\Xi}} \|\dot{\mathbf{X}} - \boldsymbol{\Theta}(\mathbf{X})\boldsymbol{\Xi}\|_2 + \lambda\|\boldsymbol{\Xi}\|_0, \tag{2.4}$$

such that the sparse vector $\boldsymbol{\xi}_i$ forms the matrix $\boldsymbol{\Xi} = [\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_n] \in \mathbb{R}^{v \times n}$ that selects the active terms in the library function $\boldsymbol{\Theta}$. The amount of sparsity promotion is controlled by the parameter λ, which determines the penalization by the ℓ_0-norm. By solving Eq. (2.4), we can identify a model of system dynamics

$$\frac{d}{dt}\mathbf{x}(t) = \boldsymbol{f}(\mathbf{x}(t)) \approx \boldsymbol{\Theta}(\mathbf{x}(t))\boldsymbol{\Xi}. \tag{2.5}$$

Many different optimization techniques can be used to obtain the sparse coefficients $\boldsymbol{\Xi}$. In the original SINDy paper, the sequential thresholded least squares (STLSQ) [180] is used to perform the sparse regression. However, other methods could also be used to solve this optimization problem as well, as we will show in Sec. 2.3. Fig. 2.1 illustrate the overall idea of the SINDy algorithm.

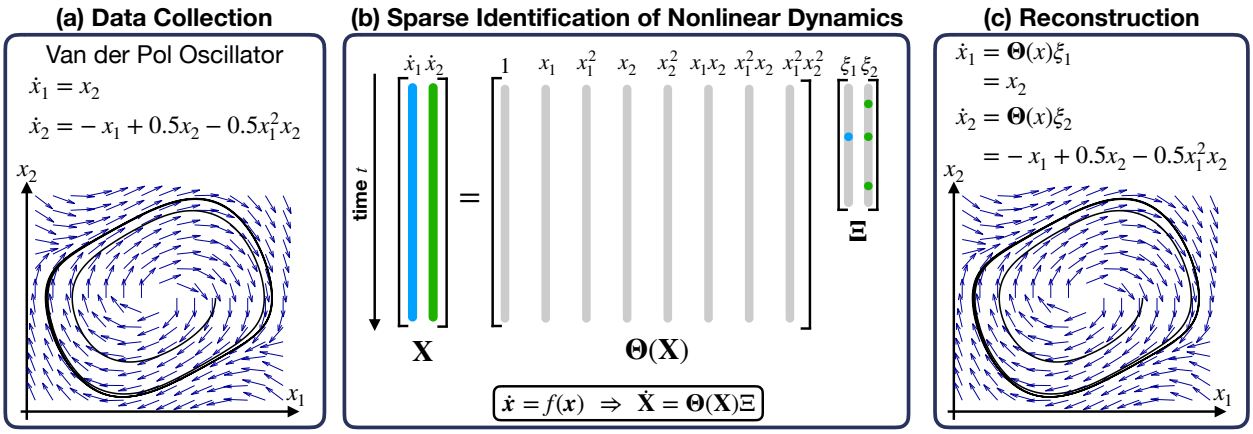


Figure 2.1: An illustration of the SINDy algorithm [180] on the Van der Pol oscillator. (a) The system state and its derivative are measured. (b) A user defined library function is constructed, and a sparse matrix of coefficients $\mathbf{\Xi}$ is solved for, representing the few active terms in the dynamics. (c) The SINDy discovered sparse model is formed and used to reconstruct the phase portrait.

2.2 Implicit SINDy Algorithm

We briefly introduce the full multidimensional implicit-SINDy algorithm, which will provide a foundation for our robust implicit identification algorithm in Sec. 3.2.

The implicit-SINDy algorithm [181] extends SINDy to identify implicit differential equations

$$\boldsymbol{f}(\mathbf{x}, \dot{\mathbf{x}}) = 0, \tag{2.6}$$

and in particular, systems that include rational functions in the dynamics, such as chemical reactions and metabolic networks that have a separation of timescales. The implicit-SINDy generalizes the library $\mathbf{\Theta}(\mathbf{X})$ in (2.5) to include functions of $\mathbf{x}$ and $\dot{\mathbf{x}}$:

$$\mathbf{\Theta}(\mathbf{X}, \dot{\mathbf{X}})\mathbf{\Xi} = \mathbf{0}. \tag{2.7}$$

However, this approach requires solving for a matrix $\mathbf{\Xi}$ whose columns $\boldsymbol{\xi}_i$ are sparse vectors in the null space of $\mathbf{\Theta}(\mathbf{X}, \dot{\mathbf{X}})$. This approach is non-convex, relying on the al-

ternating directions method (ADM) [181, 182], and null space computations are highly ill-conditioned for noisy data [59, 181, 183], thus inspiring the our work on SINDy-PI shown in Chapter. 3.

2.3 Sparsity Promoting Techniques

At its core, the original SINDy algorithm is a selection process based on thresholded iterative least-square regression. Its final goal is to select the fewest candidate functions from the library to represent the left-hand side derivative. This well-known ℓ_0 problem is nondeterministic polynomial time (NP) hard. Thus, certain relaxation is needed to solve this problem. The STLSQ is one such relaxed regression technique that promotes the sparsity of the final solutions through iterative regression. Zhang and Schaeffer et al. [184] performed a rigorous theoretical guarantee on the convergence of the original SINDy algorithm. They proved that the STLSQ approximates local minimizers of an unconstrained ℓ_0-penalized least-squares problem. They also analyzed the sufficient conditions for its convergence, rate of convergence, and requirements for one-step recovery. When the library functions or state variables are magnitudes apart, the STLSQ might not be able to identify the correct dynamics. This is a common issue in the application of biology and epidemiology [185] especially when the data set is noisy. One can use the weighted thresholding version of STLSQ shown in [185] to overcome this problem, where the new threshold value is calculated by dividing the original constant threshold λ by the norm of the corresponding column. Another practical way of avoiding this is by normalizing the library function such that each candidate function have a similar range of magnitude. However, this requires an additional step to recover the correct parameter value for the final model.

Another way to relax the original NP-hard problem is by using the least absolute shrinkage and selection operator (LASSO) approach [186, 187, 187]. The LASSO approach uses the ℓ_1 penalty on the SINDy's selection matrix to improve the sparsity of the final solution. This ℓ_1 relaxation allows tractable computations to be performed and is widely available

through many standard software packages, such as Matlab's **lasso** command, Python's Scikit-Learn package [188], etc. It is noteworthy that it is possible to include the weighting terms in the LASSO, which will result in adaptive LASSO [189]. The use of the adaptive weight can help us avoid local minimums. Other works that use weighted ℓ_1 norm to promote sparsity include [190] and [191]. In the work of [190], weights of ℓ_1 norm are adjusted to improve the regression robustness to noisy state measurement and state derivative. They also used second-order Tikhonov regularization for the derivative approximation to further robustify the sparse regression. The regression algorithm they proposed is the re-weighted version of the Basis Pursuit Denoising algorithm (WBPDN) [192]. In the work of [191], Pan et al. used an iterative re-weighted ℓ_1-minimization algorithm to propose general sparsity-inducing priors on the parameters of the system, which can be easily fused with the SINDy framework. In practice, we find out that LASSO-like algorithms show inferior performance compared with STLSQ like algorithms (hard thresholding approach). This might be caused by the fact that LASSO can not constrain the irrelevant terms as zero during the regression process. Instead, it still allows terms that are not important to have low magnitude coefficients, possibly allowing room for more parameter error.

It is also possible to include the ℓ_1 and ℓ_2 cost value in the cost function to impose sparsity on the final solutions such as [193] (elastic net (Enet)), [194] shows, or by just using the ℓ_2 norm [195–197]. In [12], the sequentially thresholded ridge regression (STRidge) is proposed, which promotes the sparsity using a combination of ℓ_0 and ℓ_2 norm. It serves as a modification of the STLSQ algorithm [198]. STRidge is better at identifying nonlinear PDE models, where highly correlated data is common. In [199], the use of ℓ_∞ norm on promoting the sparsity of the SINDy regression is studied. [199] suggests it is reasonable to use ℓ_∞ approximation rather than ℓ_2 approximation when no prior knowledge of derivative approximation error is available.

Besides STLSQ and LASSO, the SINDy framework can also use other sparsity-promoting techniques. In [200], Boninsegna et al. proposed Sparse Stepwise Regression (SSR) algorithm. Unlike STLSQ, the SSR algorithm only removes the most unlikely candidate

function (candidate function with the smallest magnitude) from the library during each iterative regression process. Then, it selects the final model by checking the Pareto front. The advantage of the SSR algorithm is that it avoids the problem of choosing the sparsity (tunning) parameter, which is needed in the LASSO approach and STLSQ. Similar to SSR, the work of [201] and [202] also increase the sparsity level of the identified equations by removing candidate term one at a time. The [202] call this process Hierarchical Ridge Regression (HRidge) though. Opposite to the SSR-like algorithm, the Orthogonal Matching Pursuit (OMP) approach [200, 203] adds library terms one at a time to generate different candidate models and then uses a Pareto front plot to perform model selection. This similar approach also appears in [204], where the weak formulation is used, and the author named their sparsity-promoting process Unified Sparse Dynamics Learning (USDL). The SINDy can be thought of as a sparse dictionary learning problem. Thus, the SINDy framework can use many sparsity-promoting selection algorithms in the dictionary learning problem. Some of the choices includes forward greedy selection algorithms [68, 205], backward greedy selection algorithm [206], and forward-backward approach [207, 208].

Group sparsity is another approach for promoting sparsity, and it helps identify nonlinear equations when: 1) Multiple experimental data are collected for the same type of experiments [209]. 2) Multiple data sets of the same ODE system or PDE system are given with different parametric value [210, 211]. Group LASSO (GLASSO) [212, 213] and sequentially grouped threshold ridge regression (SGTR) are two popular ways to promote group sparsity. On the robustness wise, the work of [211] points out SGTR is found to be more robust than GLASSO when it comes to identifying parametric PDE.

Other thresholding techniques include Sparse Relaxed Regularized Regression (SR3) [214–216], which is more robust to noisy measurement and outliers and can include physical constraints. Recently, Lu et al. proposed a non-convex penalty least-squares algorithm (NPSLS) that can be used for SINDy algorithm [217], and the author shows the NPSLS has better performance in high-noise regions than STLSQ. The sparse Bayesian learning approach [218] can also be used in the SINDy framework to promote sparsity. There are considerable

algorithms and papers regarding sparsity promoting subset selection problems, and we recommend the interested reader to read [219] for an overview. The noise sensitivity analysis of the sparsity promoting algorithm used in the SINDy framework is analyzed in the work of [220]. To be more specific, the author study and assess the performance of local and global filtering regression techniques to improve the accuracy and robustness of STLSQ and WBPDN.

Although the goal of the SINDy framework is to provide the user with a sparse and interpretable model, it is noteworthy that having a sparse solution does not necessarily mean the model is correct. The identified model is sparse when the proper basis is used to represent it, and the inverse is not valid. If the model only consists of several candidate functions, it does not necessarily mean we presented it on the correct basis. This excellent point is pointed out in the work of [221]. The work of [221] proposed the Entropic Regression approach. This approach uses an information-theoretic measure for the data-driven discovery of the underlying dynamics, and it is robust to the noise and outliers in the measurement data. The author suggests that using metric-based cost function as error quantification is a root cause of failure for many sparse regression methods under significant noise and outliers. They indicate that the outlier data points tend to deviate from the other data points when measured by metric distance. This property makes the sparse regression fails since minimizing the metric distance caused by outliers means putting much less weight on the relatively clean measurements. The Entropic Regression approach is an information-theoretic regression approach that emphasizes "information relevance" according to a model-free, entropic criterion. As a result, the basis term will only be included when relevant and not necessarily because they add up a sparse model with low cost.

Many sparsity-promoting algorithms generate a pool of candidate models. This makes the model selection a crucial step. Usually, the model is selected to satisfy one fundamental principle: the model should predict most of the dynamics with the least chosen terms. Usually, this is achieved by using some model selection criteria. In [222], the Mangan et al.

used Akaike information criteria (AIC) [223, 224] and Bayes information criteria (BIC) [225] scores for selecting the best model from a pool of candidate models. Another way to choose the model is by looking at the candidate model's performance on the testing data, or called cross validation (CV) [226].

2.4 Major Varients of SINDy Algorithm

The SINDy algorithm has many extensions, and this section aims to summarize them. Those extensions aim to improve the original SINDy algorithm's noise robustness and applicability. We organize this section by grouping those major variants into sub-sections based on their functionality. PySINDy packages is an excellent Python package that implements many of the SINDy variants mentioned in the following subsections [227, 228]. Moreover, Table. 2.1 shows the summary of major variations of SINDy algorithms mentioned in this section.

2.4.1 Robustness Improvements

The original SINDy algorithm is not noise-robust, and many extensions have been made to improve the noise robustness of the SINDy algorithm. Some of those algorithms improve the noise robustness of the SINDy framework by using cleaner data to identify a model, such as the subsampling technique [198, 229–231]. Others improve the noise robustness by avoiding taking derivatives on the noisy data set, such as weak or integral formulation [232–238]. It has also been suggested that handling the derivatives of the SINDy model internally can also improve the robustness of the spare model identification process to a new level [15, 230, 239, 240]. Some works have also focused on using new regression techniques to make the SINDy more robust. In this section, we will introduce those extensions in detail.

Subsampling technique is a promising way to greatly improve the noise robustness of the SINDy algorithm [198, 229–231]. The work of Zhang et al. [229] proposed the idea of

subsampling and used it with sparse Bayesian regression (SubTSBR) [241]. They reported that as the number of subsamples increases, the algorithm's accuracy increases. SubTSBR selects data points randomly to estimate their weight and uses the model-selection criterion to evaluate the performance of estimations. If the estimation is "good," the SubTSBR algorithm marks the selected data points as high-quality, thus obtaining the final result. This algorithm works because subsamples can help select the high-quality data with low noise, thus improving the accuracy of the final model. This subsampling approach can also help eliminate the negative effect of outliers in the data since those points are neglected during the subsampling process. This technique can be applied to SINDy to improve the noise robustness. In the work of [230], Abdullah et al. used a similar subsampling technique to increase the noise robustness of the sparse regression.

The work of [198] proposed Ensemble-SINDy (E-SINDy) and uses subsampling technique on both the data and the library. With the combination use of weak formulation [233] and ensembled technique [242–244], the E-SINDy is noise-robust. The E-SINDy combined several techniques into the SINDy framework, and they can be used with other sparsity promoting algorithms in Sec. 2.3. The first technique E-SINDy uses is bagging [242]. The bagging technique uses data bootstraps to identify an ensemble of models. Each model is generated by randomly sampling data from the original data set with replacement. Then, the mean of the model coefficients is calculated along with its distribution. By generating multiple models, one can calculate the inclusion probability of each basis function, and use this inclusion probability to promote the sparsity in the final model. The second technique used in the E-SINDy is bragging [243, 244], and this method is almost the same as the bagging method, with the difference being its inclusion probability is calculated using the median of the ensembled model instead of the mean. The third technique introduced is library bootstrapping. This technique generates an ensemble of models by using sub-libraries generated by sampling different candidate basis functions from the original large library. Then, each model produced by the sub-sampled library is used to calculate the inclusion probability of the certain candidate function. If the inclusion proba-

bility is too low, it is neglected during the next iterations. The final method introduced is active learning [245, 246]. Active learning tries to collect data from regions that maximally inform the learning process (generates more variance on the current model prediction). By using this technique, one can effectively reduce the amount of data needed to create a model [247, 248]. The active learning approach first identifies the initial E-SINDy model using a small amount of randomly selected data points. Next, random initial conditions are chosen, and the previous E-SINDy models are used to perform an ensemble forecast using those random initial conditions. Then, the initial conditions' trajectories that generate the worst variance are sampled and included in the current data set. Finally, a new E-SINDy model is generated using the updated data set, and the previously mentioned process is iterated until the model accuracy and/or variance are converged. It has been shown that E-SINDy is more efficient in learning a model than the original SINDy and thus suits better in application with NMPC [198].

In the work of [231], a toolkit for SINDy in high-noise regimes is proposed. This toolkit includes several techniques: (1) weight timepoints based on estimated noise; (2) estimating model coefficients using ensemble techniques; (3) regression using FFTs; (4) leverage linear dependence of functionals; (5) restore and protect culled functionals based on Figures of Merit (FoMs). In the first tool, the goal is to weigh all the points used in the regression and figure out which points are noisier. Those noisy points are then weighted less during the regression. In the second tool, the original data set is subsampled to generate an ensembled SINDy model, and those models are used to calculate the median of the coefficients. In the third tool, the regression is targeted to fit derivatives' complex FFT coefficients. The author suspects that "regression on the FFT power spectrum coefficients knocks out the high-frequency artifacts because ℓ_2 optimization minimizes its loss by preferentially failing on small-magnitude coefficients in order to match large-magnitude coefficients. This pushes the regression to ignore the artifacts, reducing their harmful effect" [231]. In the fourth tool, linear dependence identifies alternate candidate functionals. The final tool calculates the FoMs to determine which candidate model (with different sparsity) should be used. Some

examples of FoMs are the relative error of the standard deviation of evolved trajectory versus true trajectory, whether multiple evolutions of the candidate models match each other, etc. By using all those tools, the work of [231] shows SINDy can use used to identify sparse governing equations and coefficients from highly noisy ($50 - 300\%$) time-series data.

In the original SINDy algorithm, the user must provide the state derivative to generate a sparse model. However, when the state derivative is not directly measurable, it has to be estimated using the noisy data set through numerical differentiation, which magnifies the effect of noise. Thus, some methods try to avoid using derivative data to improve the noise robustness of the SINDy algorithm [232–238]. The work of [232] uses Weak formulation to identify PDE equations. The key benefit of the weak formulation is that it can eliminate or reduce the order of the derivative terms. The work of [232] also proposed a new way to promote the sparsity of the regression. They first used singular value decomposition to estimate the sparse solutions. Then, the candidate term with the smallest magnitude is eliminated. Finally, the residual value is calculated for this smaller library and the original library. If the residual value of the smaller library is less than the previous residual value by a constant factor, then the term with the smallest magnitude is thresholded. The author claims the advantage of this process is that it can "always identify the correct form of the governing PDE even in the presence of extremely high noise," and the tuning is much easier than sequential threshold ridge regression. In the work of [233], the Weak-SINDy (WSINDy) is used to identify the ODEs from noise measurement data. Using weak formulation, the derivative of the state measurements does not need to be approximated anymore, which significantly improves the noise robustness of the sparse regression. A similar formulation can be used to identify nonlinear partial differential equations (PDEs) as well [234]. In the recent work [235], the online version of the WSINDy is proposed for "on the fly" identification of the nonlinear PDEs, and several numerical examples are used to test the proposed algorithm.

Other than weak formulation, the integral SINDy also tries to use a similar technique

to robustify SINDy by avoiding taking derivative on the noise data [236]. However, how well it works depends on the structure of the system to be identified. In [237], the integral form of the differential equation is derived to estimate unknown amplitudes of basis functions in the sparse regression. This approach also avoids taking derivatives of the noisy data and can be generalized for identifying second and higher-order systems. This is achieved by the clever use of inverse Laplace transforms on the higher-order system. In the work of [238], Lin et al. effectively used Duhamel's integral to avoid approximating the second derivative of the measurement signal. This approach improves noise robustness, especially when the second derivative is not directly measurable and needs to be estimated by taking finite-difference twice on the original measurement data. They also used the separable least squares method to identify the linear subsystem's parameters and the nonlinear subsystem's coefficients separately. To test the effectiveness of their algorithm, they performed experiments on the high-static-low-dynamic stiffness (HSLDS) nonlinear isolator system and successfully identified its nonlinear dynamics.

Another way to avoid numerically estimating the state derivative is to evaluate the state derivative internally by simulating the right-hand side of the SINDy model [15, 230, 239, 240]. In [239], RK4-SINDy is proposed by Goyal et al. In RK4-SINDy, the 4th order Runge-Kutta numerical integration framework is fused with SINDy, which does not require derivative approximation at any stage. However, this new framework is likely non-convex, and the author proposed Fix Cutoff Thresholding Procedure and Iterative Cutoff Thresholding Procedure to solve the corresponding optimization problem. Moreover, the author extended the proposed framework to learn parameterized libraries. For example, it is possible to select the nonlinear basis $e^{\alpha x}$ while learning the correct parameters α. This work also considers learning rational systems by separately understanding the numerator and denominator. In [240], Lee et al. proposed Neural SINDy (nSINDy). The nSINDy overcomes the disadvantage of taking the derivative data in the SINDy framework by using a Neural ODE [120] like formulation. Moreover, Lee et al. extended the dictionary-based parameterization approach to structure-preserving parameterization techniques.

In the work of [230], Abdullah et al. used a co-teaching approach to improve the noise robustness of the model identification process along with subsampling technique. They named it sparse identification with subsampling and co-teaching (SISC). The co-teaching method is mainly developed in the NN application, which helps overcome the overfitting issue when the data contains high noise. The co-teaching uses noise-free data from first-principles model simulation to assist the training process [249, 250]. By combining both subsampling techniques and co-teaching, the SISC approach outperforms the original SINDy method and subsampling approach. This also improves the noise robustness by internally calculating the derivative. In the work of [15], the Modified SINDy is proposed. Modified SINDy learns the sparse dynamics and noise simultaneously. Moreover, it approximates the derivative of the data set by using right-hand side of the dynamics.

Some works focused on modifying the regression process of the SINDy to improve its noise robustness [215, 251–257]. For example, in the work of [251] and [215], the SINDy regression is modified to robustly handle the outliers in the data. In the work of [251], Tran and Ward showed that SINDy could identify nonlinear systems as long as the outliers data points are sufficiently less than clean data. Moreover, they modified the SINDy algorithm to include Lagrangian/Bregman distance and used an alternating minimization approach to identify the nonlinear dynamics with measurement data that contains outliers. Later, the work of [215] relaxed the optimization problem in the SINDy regression, and allows the robust handling of outliers and corrupt data. In the work of [257], the Levenberg–Marquardt (LM) algorithm [258, 259] is used with SINDy to improve the prediction accuracy and overcome the over-fitting issue of the SINDy library. In this framework, the SINDy is first performed to determine an initial guess of the sparsity matrix. Then the LM algorithm is used to update the sparsity matrix to improve accuracy and robustness of the algorithm.

The work of [252] also robustfy SINDy by proposing sensitivity analysis SINDy (SINDy-SA). The SINDy-SA method calculates the sensitivity of the basis function parameter by Elementary Effects (EE) method [260]. At each iteration of the SINDy-SA algorithm, a

least-squares solution with ℓ_2 regularization is first solved. Next, the estimated model and numerical derivative of the measurement data are used to calculate the sum of squared errors (SSE). Then sensitivity analysis is performed to determine which basis to exclude if not all the basis function is important. The basis function is eliminated one by one, and after each elimination, the new SSE is calculated. If the SSE increases too much, it suggests the sparsest model has been found. Else, the sensitivity analysis is performed, and the algorithm starts a new iteration unless all the basis functions are determined as important during the regression process. The benefit of the SINDy-SA algorithm is that it avoids the swiping of the threasholding (tunning) parameters in the SINDy.

The work of [253] proposed an alternative framework of SINDy that uses the Conditional Gradient (CG) algorithm to identify the nonlinear dynamics, resulting in the name CINDy. The advantage of using the CG algorithm is that it serves as a different way to promote the sparsity besides using the ℓ_1 norm. It also allows the natural inclusion of linear equality and inequality constraints, reflecting symmetry or conservation assumptions for a specific problem, thus making the algorithm more robust. This reflects the idea that when the partial knowledge of the system is known, it is possible to include that knowledge into the regression framework via constraints and make the model identification more robust. For example, the SINDy-SR3 can also impose physical constraints for the regression problem and thus allows the robust identification of desired dynamics[215].

Some works tried to robustify the SINDy algorithm by including constraints in the SINDy regression [254, 255]. In the original SINDy framework, no constraint on the identified dynamical system is proposed, making it possible to lose existing symmetries in the governing equations. Thus, in the work of [254], Loiseau et al. proposed sparse Galerkin regression to include physical constraints. Those constraints can enforce energy-preserving nonlinearities or symmetries in the identified equations. Shear-driven cavity flow and cylinder flow are used to test their proposed framework. The work of [255] introduces the trapping-SINDy. The trapping-SINDy modifies the original SINDy cost function to identify a sparse model and bias the identified model to fulfill global stability

criteria. Thus, the discovered model produces long-term bounded simulation with no extra assumptions about the stability properties of equilibrium points and equilibrium trajectories. It is also possible to include multiple machine learning techniques to improve the noise robustness as the work of França et al. [261] shows. França et al. [261] proposed SINDyFE (feature engineering) that uses several machine learning techniques to improve the noise robustness of SINDy, such as sparse regression, information criteria, feature selection, and dimensionality reduction.

Other works use the Bayesian approach to include the uncertainty information of the model parameters and thus robustify the SINDy [241, 256]. In the work of [256], the variants of the SINDy algorithm that accounts for uncertainty quantification is proposed (UQ-SINDy). The UQ-SINDy uses sparse Bayesian inference to quantify the uncertainty of the SINDy coefficients. Moreover, it provides the probability of inclusion for each candidate function. Two sparsity prior spike-and-slab prior [262–264] (corresponds to ss-SINDy), and the regularized horseshoe (corresponds to rh-SINDy) prior [265] are used to generate a sparse model. Although this work is robust against noise and sparse time-series data, it can be computationally heavy for large SINDy libraries since it samples high-dimensional posterior distributions using Markov Chain Monte Carlo (MCMC).

2.4.2 Extensions to Different Applications

The original SINDy algorithms mainly focus on identifying the time-invariant first-order ODEs. However, some systems exhibit much more complicated dynamics, and the SINDy algorithm needs to be modified to discover their models. Thus, this section summarizes some of the most important extensions to the SINDy algorithm that identify different types of dynamics. For example, instead of identifying ODEs, the SINDy can be extended to identify the PDEs as well [12, 211, 232, 234, 235].

When the dynamics of interest is the time-varying system, the work of [195, 266–268] can be used. In [195], abrupt-SINDy is proposed to identify the dynamical system that

might undergo any abrupt change. In this framework, the abrupt change is detected by comparing the estimated Lyapunov time of the data with the model prediction. Then, SINDy updates the identified model by adding, deleting, or modifying existing model terms. Similarly, the work of Li et al. [266] identifies time-varying nonlinear dynamics through SINDy, where time-windowed data is collected to estimate the change in the model parameters. In [267], the Hybrid SINDy is proposed, which can model the hybrid dynamics using the clustering approach on the measurement data. This approach helps model the piece-wise nonlinear dynamics. The work of [268] achieves a similar task, where the SINDy algorithm that identifies hysteresis-controlled devices is proposed.

Sometimes, the dynamics of interest is implicit, and the work of [177, 181, 239, 241] canbe used to identify rational or implicit dynamics. In [181], Mangan et al. proposed a new way to build a library matrix that allows the combination of state derivatives and state measurement, which made the identification of implicit dynamics possible. The author called their algorithm implicit-SINDy. However, the implicit-SINDy algorithm is not robust to noise since it needs to calculate the null space of the library matrix, which is not guaranteed to exist when there's noise in the measurement data. The work of [177, 241] proposed a new way to identify the implicit equations. Both of the work bypasses the need to calculate the null space. The result is a much more robust way to determine the implicit or rational equations. The algorithm proposed in [177] is named SINDy-PI, where PI stands for parallel and implicit.

Some dynamical system includes fast and slow dynamics, and the identification of both the fast and slow dynamics is desired. In the work of [269], the problem of identifying multi-scale nonlinear systems using SINDy is considered. Bramburger et al. first used DMD to identify fast dynamics, then used SINDy to identify slow dynamics. Besides identifying fast and slow dynamics, the SINDy can also be modified to identify ROM. In the work of [196, 270], the CNN-SINDy is proposed to learn the nonlinear reduced-order model to extract low-dimensional modes and to predict their temporal evolutions. The autoencoder-type Convolutional Neural Network (CNN) can learn the nonlinearity of data.

It extracts low-dimensional modes, and then SINDy is used to obtain CNN's temporal evolution of mapped data. Thus it is named as CNN-SINDy. If the identification of the reaction work is necessary, the work of [194] can be used. In the work of [194], the reactive-SINDy is proposed to estimate a parsimonious reaction network. The main difference between the reactive SINDy and SINDy is that it uses the vector-valued ansatz function to build a library. Moreover, its loss functions includes both ℓ_1 and ℓ_2 loss.

Identifying the partial dynamics instead of the entire system dynamics sometimes relaxes the difficulties of building the correct library. In some cases, a model might be already available, and the goal is to explain the difference between the current model and the measurement. This is the problem of discrepancy modeling, and SINDy can also be used to identify the missing dynamics instead of identifying the whole dynamics of the system as [178, 271] shows. The work of [178] mainly considers using SINDy to identify dynamics discrepancy, while the work of [271] extends SINDy to identify the systematic state-space residual as well. Moreover, the work of [271] also considers other approaches such as GR, NN, and DMD to obtain discrepancy models. It also demonstrates the impact of noise on the successful disambiguation between the deterministic and random effects.

A common issue in the SINDy framework is the choice of the library function. Usually, the polynomial or trigonometric or the combination of both is used to identify a nonlinear model. However, this does not guarantee that the library includes all the candidate functions needed to recover the nonlinear dynamics (closure issue). This is when the expert knowledge comes in handy. When there's no expert knowledge available, using an evolutionary-based approach is a good way to build up the library and extend the space of nonlinear functions as [272, 273] shows. Due to the sparse regression process, this approach is faster than the genetic algorithm. However, its drawback is that the GP algorithm still needs basic building blocks to be included [273]. The author of [273] tested this idea on the experimental data of the Duffing oscillator, and it shows good performance. Another common problem of the SINDy framework is that it requires full-state measurement data. This issue is recently discussed in the work of Bakarji et al. [274]. Table. 2.1 summarizes

the major variants of SINDy mentioned in this section.

Table 2.1: This table shows the summary of major variants/modifications of the SINDy algorithms.

Variants' Name or **Sources**	**Applications/Capabilities**
PDE-Find [12, 211]	Identifies nonlinear PDEs.
Abrupt-SINDy [195]	Identifies dynamical models that undergo abrupt change.
Li et al. [266]	Identifies time-varying nonlinear dynamics.
Implicit-SINDy [181]	Identifies implicit ODEs by calculating null space.
SINDy-PI [177]	Identifies implicit ODEs, PDEs, and physical laws.
Zhang et al. [241]	Identifies implicit ODEs, PDEs and physical laws through Bayesian approach.
SINDy-SR3 [215]	1. Robust handeling of outlier and corrupted data. 2. Parametric dependencies in candidate library functions. 3. Impose physical constraints.
Bramburger et al. [269].	Identifies multi-scale nonlinear systems using SINDy.
Neural SINDy (nSINDy) [240]	1. nSINDy avoids taking derivatives of noisy data. 2. It uses a Neural ODE-like formulation to improve the noise robustness.
SINDyFE (feature engineering) [261]	Uses multiple machine learning techniques to improve the noise robustness of SINDy.
Hybrid SINDy [267]	Identifies piece-wise nonlinear dynamics.
Loiseau1 et al. [254]	1. Uses sparse Galerkin regression to include physical constraints. 2. Those constraints can enforce energy-preserving nonlinearities or symmetries in the identified equations.
UQ-SINDy [256]	Uses sparse Bayesian inference to quantify the uncertainty of the SINDy coefficients.
Trapping-SINDy [255]	Discover sparse model that produces long-term bounded simulation.
CNN-SINDy [196, 270]	Identifies nonlinear reduced-order model.

Table 2.1 continued from previous page

Tran and Ward et al. [251]	Identifies nonlinear sparse model from data that contains outliers.
Reactive-SINDy [194]	Estimates a parsimonious reaction network.
Jiang et al. [257]	1. Uses LM algorithm with SINDy to improve the prediction accuracy. 2. Overcomes the over-fitting of the SINDy library.
Weak-SINDy [233, 234]	Robust identification of nonlinear ODEs and PDEs using weak formulation.
Messenger et al. [235]	Online identification of nonlinear PDEs.
Guého et al. [237]	Identifies nonlinear ODEs using integral formulation.
RK4-SINDy [239]	1. Identifies nonlinear ODEs using the 4th order Runge-Kutta numerical integration framework. 2. Identifies rational ODEs. 3. Allows the construction of parameterized library.
Gurevich et al. [232]	1. Uses weak formulation to identify nonlinear PDEs. 2. Proposed sparsity promoting algorithms using SVD.
Integral-SINDy [236]	Identifies nonlinear ODEs using integral formulation.
Zhang et al. [229]	1. Identifies nonlinear ODEs with subsampling and sparse Bayesian regression (SubTSBR). 2. Increased the noise robustness of the identification.
Abdullah et al. [230]	Robustly identifies nonlinear ODEs with subsampling and co-teaching approach.
Lin et al. [238]	1. Uses Duhamel's integral to avoid solving the second derivative of the measurement signal. 2. Improved the noise robustness of the identification process when second derivative is needed.
Quade et al. [272]	Discussed the possibility of using GP with SINDy.
Goharoodi et al. [273]	Combined SINDy with GP.
SINDy-SA [252]	Promotes sparsity using sensitivity analysis approach.
Ensemble-SINDy [198]	Robustly identifies ODEs by using bagging, bragging, and active learning.

Table 2.1 continued from previous page

Delahunt et al. [231]	1. Identifies ODE model from highly noisy ($50-300\%$) time-series data. 2. Uses multiple tools to improve the noise robustness of SINDy regression.
CINDy [253]	Identifies nonlinear ODEs using the Conditional Gradient (CG) algorithm.
SINDy-Hybrid [268]	Identifies the model of hysteresis-controlled system.
Modified-SINDy [15]	Simultaneously learn the sparse dynamics and noise.

2.5 Applications of SINDy Algorithm and its Variants

The SINDy algorithm and its variants have broad applications. For example, the SINDy algorithm and its variants have successful applications in engineering [266, 275–279]. In [266], SINDy is used to discover time-varying aeroelastic models of a long-span suspension bridge. In [275], SINDy is used to model the dynamics of radio frequency (RF) power amplifier (PA). The SINDy identify model has shown good performance and is more sparse than existing RF PA models. In [276], a SINDy-like regression framework is proposed to perform physics-informed sparse identification for full-field structural vibration tracking and analysis. Moreover, the sparsity is promoted using OMP (the term physics inform in this work refers to some prior knowledge of the underlying dynamics is given). In the work of [277], the SINDy is used to identify nonlinear optical communication systems, and Sorokina et al. named it Sparse Identification for Nonlinear Optical communication systems (SINO) method. In [278], the sparse regression is used to identify the failure criteria of composites. In [279], the variant of SINDy is used to identify the hysteresis-controlled pump system.

Closely related to the engineering field, the SINDy and its variants have successful applications in controls [280–283] and robotics [178, 284]. For example, SINDy can be used with nonlinear model predictive control (NMPC) to perform model identification and control in the low data-limit [281]. In the work of [282], the SINDy is used with MPC in

the numerical study of the longitudinal missile. Another work that combines the SINDy with NMPC is conducted in [283], where the SINDy is used to identify the models of two-time-scale processes and then used with NMPC for control purposes. In the work of [284], the SINDy is used to identify the jacobian Matrix of a 2-DoF (degrees of freedom) Agile Eye robot that performs spherical motion. In the work of [178], the SINDy is used to model the energy dissipation of the double pendulum using the discrepancy modeling approach, and a numerical example of the double pendulum is also shown to illustrate how discrepancy modeling using SINDy can improve the controller performance.

The SINDy and its variant also have broad applications in the dynamical system, mathematics, and numerical analysis [129, 274, 285–289]. In [285], the SINDy algorithm is used to identify the governing models of spatially-dependent boundary value problems (BVPs). In [286], the SINDy is combined with bootstrapping sampling approach to model the nonlinear time-delayed dynamics. In [129], the SINDy is used to identify Koopman observable functions from a pool of candidate functions. In [287], Thaler et al. proposed a SINDy-like framework, Sparse Identification of Truncation Errors (SITE), to identify truncation errors. In [288], the SINDy is used to model the Poincaré map. In the work of [274], the variant of the SINDy is used to identify a nonlinear model of a chaotic water wheel using video data. Only partial measurements are available in this case. In [289], an experimental set-up that represents the Duffing oscillator is built, and SINDy is used to identify its underlying dynamics using noisy measurement data.

The SINDy and its variants are also used in fluid flow study [196, 290, 291], mean-filed theory [292], electroconvection [293], plasma physics [201], magnetohydrodynamic (MHD) systems [173, 294], modeling waves of rotating detonation engine [295]. It can even generate a ROM for high dimensional systems [196, 270, 296, 297]. In [196], SINDy is used with convolutional neural network-based autoencoder (CNN-AE) to model low-dimensionalized complex flow phenomena. This work and the work of [83, 298] tackled the difficulty of SINDy in identifying high-dimensional systems. In [290], the SINDy is used to identify the Wagner function, which represents the response in the lift on

an airfoil that is subject to a sudden change in conditions. In [291], the SINDy-c [280] is used to identify the ROM for aerodynamic loads of oscillating airfoil. In [292], the variant of SINDy is used to identify mean-field equations from particle data. In [293], Guan et al. used SINDy to model the chaotic time evolution of the coherent structures of electroconvection data. Using POD, the coherent structures of electroconvection data are extracted. Then the SINDy is used to model the nonlinear chaotic time evolution of those coherent structures. Moreover, constraints are added during the regression to preserve the symmetries observed in the original system. In the work of [201], the variant of the SINDy is used to identify the reduced plasma physics models from fully-kinetic simulations. In the work of [173, 294], the SINDy is modified to respect the global conservation laws and is used in the modeling of magnetohydrodynamic (MHD) systems. In [296], the SINDy is used to create interpretable ROM from time-series data of multiple temperature nodes and factors influencing temperature. In [297], SINDy is used in a general dynamic reduced-order modeling framework for typical experimental data (time-resolved sensor data and optional non-time-resolved PIV snapshots). In the work of [196, 270] the CNN-SINDy is used to learn the nonlinear reduced-order model. In the work of [202], the variant of the SINDy is used to identify the models of active nematic hydrodynamics using data from experiments on microtubule-based active nematics.

Other applications of the SINDy and its variants include chemistry [299], astronomy [237, 269], biology [194, 256, 300], and epidemiology [185, 257, 301]. In [299], the Bhadriraju et al. proposed an interesting way to combine the SINDy and Deep Neural Network (DNN) to identify models for chemical processes. They call their approach operable adaptive sparse identification of systems (OASIS), which can adapt to rapidly changing dynamics. In their work, the DNN is used to predict the sparsity parameters of the SINDy model in the online process. The training data for the DNN is generated offline using the original SINDy algorithm on multiple data sets. In the work of [269], Bramburger et al. identified slow and fast dynamics of the simulated Sun-Jupiter three-body planetary model using SINDy. In [237], the sparse regression framework is investigated for the astro-

dynamics problem, and two-body Keplerian dynamics is considered an example. In [300], the SINDy is used to identify a predator-prey type reduced model from simulation data of a convection problem. In the work of [194], the variant of the SINDy is used to identify the gene regulation network. In the work of [256], the variant of SINDy (UQ-SINDy) is used to identify a model of lynx and hare populations from the real-world dataset. In [185], the SINDy is used to identify epidemiological models from both simulation and empirical data. In the work of [257], the variant of the SINDy (SINDy-LM) is used to model the transmission dynamics of COVID-19. In [301], SINDy is used to discover governing equations of IPF (idiopathic pulmonary fibrosis) disease progression.

Chapter 3

ROBUST PARALLEL IMPLICIT SPARSE IDENTIFICATION OF NONLINEAR DYNAMICS

3.1 Introduction

The sparse identification of nonlinear dynamics (SINDy) algorithm [180] discovers parsimonious models through a sparsity-promoting optimization to select only a few model terms from a library of candidate functions. SINDy has been widely adopted in the community [184, 194, 195, 200, 209, 241, 254, 267, 277, 287, 297, 298, 300, 302–305], but it relies on the dynamics having a sparse representation in a pre-defined library, making it difficult to discover implicit dynamics and rational functions. The implicit-SINDy extension [181] makes it possible to identify these implicit functions, although this algorithm is extremely sensitive to noise. In this chapter, we develop a robust, parallel algorithm for the sparse identification of implicit dynamics, making it possible to explore entirely new classes of systems that were previously inaccessible.

Parsimonious modeling has a rich history, with many scientific advances being argued on the basis of Occam's razor, that the simplest model is likely the correct one. SINDy exemplifies this principle, identifying a potentially nonlinear model with the fewest terms required to describe how the measurement data changes in time. The basic idea behind SINDy may be illustrated on a one-dimensional system $\dot{x} = f(x)$; the general formulation for multidimensional dynamics will be described in the following sections. An interpretable form of the nonlinear dynamics may be learned by writing the rate of change of the state of the system x as a sparse linear combination of a few terms in a library of

candidate functions, $\boldsymbol{\Theta}(x) = \begin{bmatrix}\theta_1(x) & \theta_2(x) & \dots & \theta_p(x)\end{bmatrix}$:

$$\dot{x}(t) = f(x(t)) \approx \boldsymbol{\Theta}(x(t))\boldsymbol{\xi}. \tag{3.1}$$

where each $\theta_j(x)$ is prescribed candidate term (e.g. $x, x^2, \sin(x), \cdots$). The derivative of the state and the library of candidate functions may both be computed from measured trajectory data. It then remains to solve for a sparse vector $\boldsymbol{\xi}$ with nonzero entries ξ_j indicating which functions $\theta_j(x)$ are active in characterizing the dynamics. The resulting models strike a balance between accuracy and efficiency, and they are highly interpretable by construction. In a short time, the SINDy algorithm has been extended to various applications as shown in Sec. 2.4 and Sec. 2.5.

The generalized linear model in (3.1) does not readily lend itself to representing implicit dynamics and rational functions, which are not naturally expressible as sum of a few basis functions. Instead, the implicit-SINDy algorithm [181] reformulates the SINDy problem in an implicit form:

$$\boldsymbol{\Theta}(x, \dot{x})\boldsymbol{\xi} = 0. \tag{3.2}$$

This formulation is flexible enough to handle a much broader class of dynamics with rational function nonlinearities, such as $\dot{x} = N(x)/D(x)$ which may be rewritten as $\dot{x}D(x) + N(x) = 0$. However, the sparsest vector $\boldsymbol{\xi}$ that satisfies (3.2) is the trivial solution $\boldsymbol{\xi} = \mathbf{0}$. Thus, the implicit-SINDy algorithm leverages a recent non-convex optimization procedure [182, 306] to find the sparsest vector $\boldsymbol{\xi}$ in the null space of $\boldsymbol{\Theta}(x, \dot{x})$, which differs from other approaches [307, 308] that identify the rational dynamics. For even small amounts of noise, the dimension of the null space will become prohibitively large, making this approach extremely sensitive to noise and compromising the model discovery process.

This work develops an optimization and model selection framework that recasts implicit-SINDy as a convex problem, making it as noise robust as the original non-implicit

SINDy algorithm and enabling the identification of implicit ODEs and PDEs that were previously inaccessible. The key to making the implicit-SINDy algorithm robust is the realization that if we know even a single term in the dynamics, corresponding to a non-zero entry ξ_j, then we can rewrite (3.2) in a non-implicit form

$$\theta_j(x, \dot{x}) = \boldsymbol{\Theta}'(x, \dot{x})\boldsymbol{\xi}' \tag{3.3}$$

where $\boldsymbol{\Theta}'$ and $\boldsymbol{\xi}'$ have the j-th element removed. Because none of these terms are known *a priori*, we sweep through the library, term by term, testing (3.3) for a sparse model that fits the data. This procedure is highly parallelizable and provides critical information for model selection. Our approach is related to the recent work of Zhang et al. [241], which also makes the implicit problem more robust by testing candidate functions individually. However, there are a number of key differences in the present approach. Our work explicitly considers *rational* nonlinearities to discover exceedingly complex implicit PDEs, such as a simplified model of the Belousov-Zhabotinsky (BZ) reaction. Our framework also provides several new greedy algorithms, including parallel and constrained formulations. We further extend this method to include the effect of control inputs, making it applicable to robotic systems [9], and we use this procedure to discover Hamiltonians. Finally, our approach provides guidance on model selection, a comprehensive comparison with previous methods, and a careful analysis of noise robustness.

3.2 SINDy-PI: Robust Parallel Identification of Implicit Dynamics

We have developed the SINDy-PI (parallel, implicit) framework for the robust identification of implicit dynamics, bypassing the null space approach discussed in Sec. 2.2. The idea is that if even a single term $\theta_j(\mathbf{x}, \dot{\mathbf{x}}) \in \boldsymbol{\Theta}(\mathbf{x}, \dot{\mathbf{x}})$ in the dynamics (2.6) is known, it is possible to rewrite (2.7) as

$$\theta_j(\mathbf{X}, \dot{\mathbf{X}}) = \boldsymbol{\Theta}(\mathbf{X}, \dot{\mathbf{X}}|\theta_j(\mathbf{X}, \dot{\mathbf{X}})\boldsymbol{\xi}_j, \tag{3.4}$$

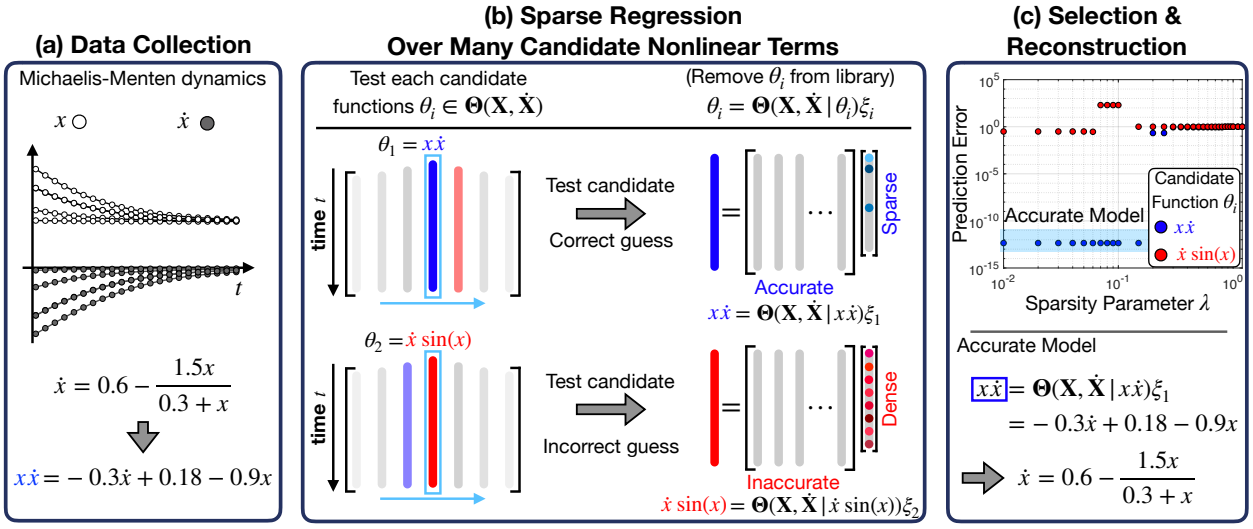


Figure 3.1: The illustration of the SINDy-PI algorithm on Michaelis-Menten dynamics. (a) The Michaelis-Menten system is simulated, and measurement data is provided to SINDy-PI. (b) Multiple possible left-hand side functions are tested at the same time. (c) The candidate model prediction error is calculated, and the best model is selected.

where $\Theta(\mathbf{X}, \dot{\mathbf{X}}|\theta_j(\mathbf{X}, \dot{\mathbf{X}}))$ is the library $\Theta(\mathbf{X}, \dot{\mathbf{X}})$ with the θ_j column removed. Equation (3.4) is no longer in implicit form, and the sparse coefficient matrix corresponding to the remaining terms may be solved for using previously developed SINDy techniques [12, 180, 191, 200, 215, 216, 236, 241, 302, 309]. In particular, we solve for a sparse coefficient vector $\boldsymbol{\xi}_j$ that minimizes the following loss function:

$$\|\theta_j(\mathbf{X}, \dot{\mathbf{X}}) - \Theta(\mathbf{X}, \dot{\mathbf{X}}|\theta_j(\mathbf{X}, \dot{\mathbf{X}})\boldsymbol{\xi}_j\|_2 + \beta \left\|\boldsymbol{\xi}_j\right\|_0, \tag{3.5}$$

where β is the sparsity promoting parameter. There are numerous relaxations of the non-convex optimization problem in (3.5), for example the sequentially thresholded least-squares algorithm [180]. Because there is no null space calculation, the resulting algorithm is considerably more robust to noise than the implicit-SINDy algorithm [181], i.e. we longer have to deal with an ill-conditioned null space problem.

In general, the entire point of SINDy is that the dynamics are not known ahead of time, and so it is necessary to test each candidate function θ_j until one of the models in (3.4)

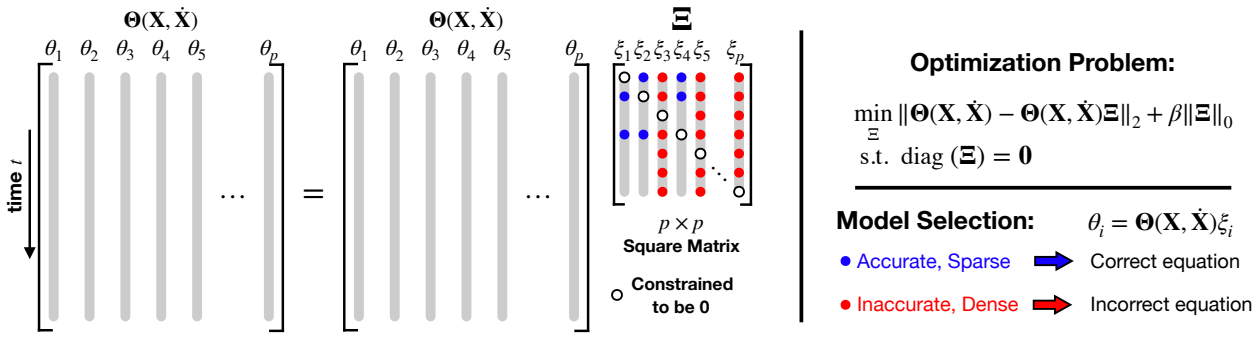


Figure 3.2: Schematic illustrating the constrained formulation of the SINDy-PI algorithm.

admits a sparse and accurate solution. When an incorrect candidate term is used, then the algorithm results in a dense (non-sparse) model $\boldsymbol{\xi}_j$ and an inaccurate model fit, and when a correct term is included, the algorithm identifies a sparse model $\boldsymbol{\xi}_j$ and an accurate model fit. In this way, it is clear when the algorithm has identified the correct model. Moreover, there is a wealth of redundant information, since each term in the correct model may be used as the candidate function on the left hand side, and the resulting models may be cross-referenced. This approach is highly parallelizable, and each candidate term may be tested simultaneously in parallel. The non-parallel formulation in (3.4) was recently introduced by Zhang et al. [241] in the context of Bayesian regression, where they also make the implicit problem more robust by testing candidate functions individually; however, they do not consider dynamics with rational function nonlinearities or control inputs. In this work, we extend the robust implicit formulation to identify several challenging implicit ODE and PDE systems with rational function nonlinearities, which are ubiquitous in engineering and natural systems, and systems with external forcing and control inputs. We also introduce the parallel formulation and model selection frameworks. Further, we will introduce a constrained optimization framework to simultaneously test all candidate functions.

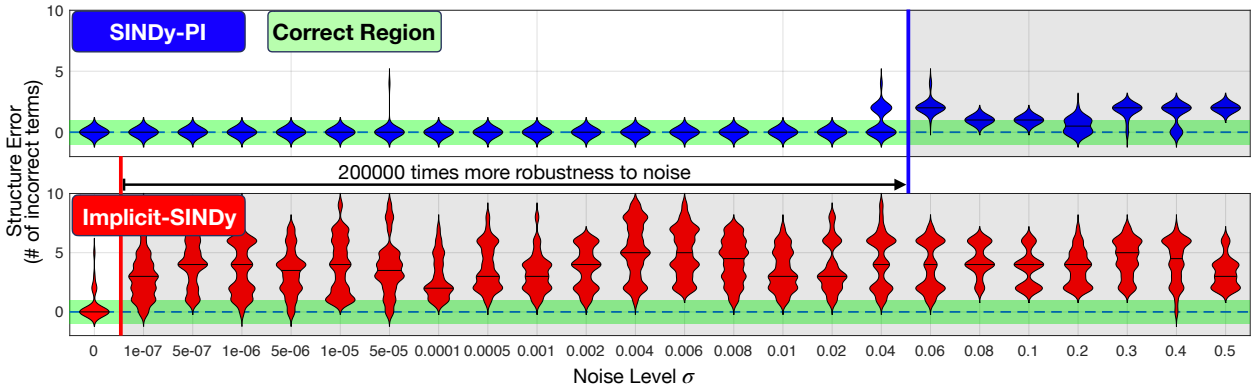


Figure 3.3: SINDy-PI and implicit-SINDy are compared on the Michaelis-Menten kinetics, where the structure error quantifies the number of terms in the model that are incorrectly added or deleted, compared with the true model. The derivative is computed by the total-variation regularization difference (TVRegDiff) [310] on noisy state measurements. The violin plots show the cross-validated distribution of the number of incorrect terms across 30 models. The green region indicates no structural difference between the identified model and the ground truth model. Details are provided in Appendix A.1.2.

3.2.1 Model Selection

For each candidate function in (3.4), we obtain one candidate model. When the candidate function θ_j is not in the true dynamics, then the resulting coefficient vector $\boldsymbol{\xi}_j$ will not be sparse and there will be large prediction error. In contrast, when a correct candidate function is selected, then we obtain a sparse coefficient vector $\boldsymbol{\xi}_j$ and small prediction error. For an implicit dynamical system, there may be several different implicit equations that must be identified, resulting in several candidate functions that admit sparse models. The sequentially thresholded least squares (STLSQ) algorithm that we use here, and whose convergence properties are considered by Zhang and Schaeffer [184], iteratively computes a least-squares solution to minimize $\|\theta_j(\mathbf{X}, \dot{\mathbf{X}}) - \boldsymbol{\Theta}(\mathbf{X}, \dot{\mathbf{X}}|\theta_j(\mathbf{X}, \dot{\mathbf{X}}))\boldsymbol{\xi}_j\|_2$ and then zeros out small entries in $\boldsymbol{\xi}_j$ that are below a set threshold λ. This threshold λ is a hyperparameter that must be tuned to select the model that most accurately balances accuracy and efficiency. Thus, we must employ model selection techniques to identify the implicit models that best supports the data, while remaining as simple as possible.

There are several valid approaches to model selection. To select a parsimonious yet accurate model we can also employ the Akaike information criterion (AIC) [223, 224] and Bayesian information criterion (BIC) [225], as in [222]. It is also possible to sweep through the parameter λ and candidate functions θ_j, and then choose the Pareto optimal model from a *family* of models on the Pareto front balancing accuracy and efficiency; this is the approach in the original SINDy work [180] and in earlier work leveraging genetic programming to discover dynamics [13, 14]. In this work, we take a different approach, selecting models based on performance on a test data set $\mathbf{X}_t$ that has been withheld for model validation to automate the model selection process. For each threshold λ, the resulting model is validated on the test set $\mathbf{X}_t$, and the model with the lowest test error is selected. One error function is the model fit:

$$\text{Error} = \frac{\left\| \theta_j(\mathbf{X}_t, \dot{\mathbf{X}}_t) - \mathbf{\Theta}(\mathbf{X}_t, \dot{\mathbf{X}}_t | \theta_j(\mathbf{X}_t, \dot{\mathbf{X}}_t)) \mathbf{\Xi} \right\|_2}{\left\| \theta_j(\mathbf{X}_t, \dot{\mathbf{X}}_t) \right\|_2}. \tag{3.6}$$

In practice, for rational dynamics, we select based upon the predicted derivative $\dot{\mathbf{X}}_t$:

$$\text{Error} = \frac{\left\| \dot{\mathbf{X}}_t - \dot{\mathbf{X}}_t^{\text{model}} \right\|_2}{\left\| \dot{\mathbf{X}}_t \right\|_2}. \tag{3.7}$$

For implicit dynamics where each state derivative may be written as a rational function

$$\dot{x}_k = f_k(\mathbf{x}) = \frac{N_k(\mathbf{x})}{D_k(\mathbf{x})}, \tag{3.8}$$

then we restrict the candidate functions to $\theta_j(\mathbf{x}, \dot{\mathbf{x}}) = \dot{x}_k \theta_j(\mathbf{x})$ for some $\theta_j(\mathbf{x}) \in \mathbf{\Theta}(\mathbf{x})$ to identify a separate sparse model for each $\dot{x}_k$. Several candidate functions may provide accurate and sparse models. These different models may further be cross-references to check that the same terms are being selected in each model, providing additional information for model selection and validation.

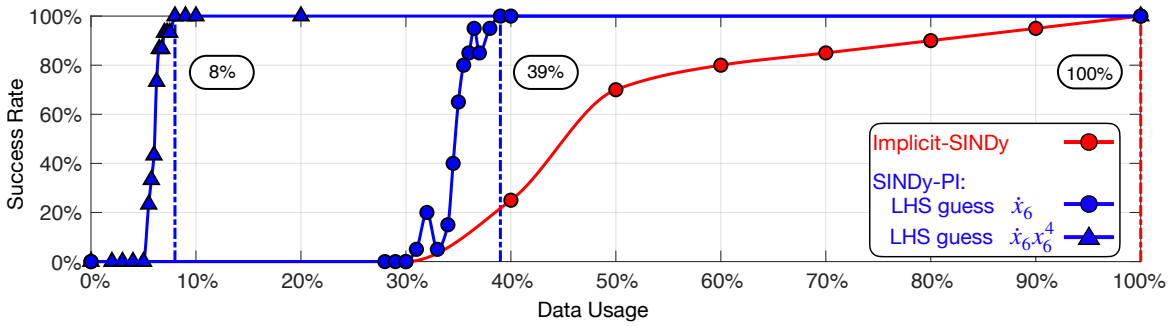


Figure 3.4: Success rate of SINDy-PI and implicit-SINDy identifying yeast glycolysis (3.13f) with different percentage of training data. Each data usage percentage is randomly sampled from the entire data set composed of all trajectories. The success rate is calculated by averaging the results of 20 runs.

3.2.2 *Constrained Optimization Formulation*

In (3.4) each candidate function was tested individually in a parallel optimization. However, each of these individual equations may be combined into a single constrained system of equations

$$\boldsymbol{\Theta}(\mathbf{X}, \dot{\mathbf{X}}) = \boldsymbol{\Theta}(\mathbf{X}, \dot{\mathbf{X}})\boldsymbol{\Xi} \quad \text{such that} \quad \boldsymbol{\Xi}_{jj} = 0. \tag{3.9}$$

We constrain $\boldsymbol{\Xi}$ to have zero entries on the diagonal, as shown in Fig. 3.2, which is the same as removing the candidate function from the library in the separate optimization problems in (3.4). Without this constraint, the trivial solution $\boldsymbol{\Xi} = \mathbb{I}_{p\times p}$ will provide the sparsest $\boldsymbol{\Xi}$ and the most accurate model. This may be written as a formal constrained optimization problem:

$$\begin{aligned} \min_{\boldsymbol{\Xi}} \quad & ||\boldsymbol{\Theta}(\mathbf{X}, \dot{\mathbf{X}}) - \boldsymbol{\Theta}(\mathbf{X}, \dot{\mathbf{X}})\boldsymbol{\Xi}||_2 + \beta\|\boldsymbol{\Xi}\|_0, \\ \text{s.t.} \quad & \text{diag}(\boldsymbol{\Xi}) = \mathbf{0}. \end{aligned} \tag{3.10}$$

This optimization is non-convex, although there are many relaxations that result in accurate and efficient proxy solutions [180, 187, 216]. In this work, we will use sequentially thresholded least squares, so that any entry $\boldsymbol{\Xi}_{ij} < \lambda$ will be set to zero; the sparsity parameter

λ is a hyperparameter, and each column equation may require a different parameter λ_j. The constrained formulation in (3.10) can be solved efficiently in modern optimization packages, and we use CVX [311, 312]. After solving (3.10) we have numerous candidate models, one for each column $\boldsymbol{\xi}_k$ of $\boldsymbol{\Xi}$, given by

$$\boldsymbol{\Theta}(\mathbf{X}, \dot{\mathbf{X}})\boldsymbol{\xi}_j = 0. \tag{3.11}$$

The sparse models that result in an accurate fit are candidate implicit models, and they may be assessed using the model selection approaches outlined above. These various models may be cross-referenced for consistency, as the same models will have the same sparsity pattern. This information can then be used to refine the library $\boldsymbol{\Theta}$, for example to only include the nonzero entries in the sparse columns of $\boldsymbol{\Xi}$.

3.2.3 Noise Robustness

We now compare the noise sensitivity of SINDy-PI and implicit-SINDy on the one-dimensional Michaelis–Menten model for enzyme kinetics [181, 313, 314], given by

$$\dot{x} = j_x - \frac{V_{\max} x}{K_m + x}, \tag{3.12}$$

where x denotes the concentration of the substrate, j_x denotes the influx of the substrate, V_{max} denotes the maximum reaction time, and K_m represents the concentration of half-maximal reaction. We use the same parameters as in [181], with $j_x = 0.6$, $V_{max} = 1.5$, and $K_m = 0.3$. Figure 3.3 shows the result of the noise robustness of SINDy-PI and implicit-SINDy. In this example, SINDy-PI is able to handle over 10^5 more measurement noise than implicit-SINDy, while still accurately recovering the correct model. Details are provided in Appendix. A.1, and key factors that limit robustness are discussed inn Appendix. A.9.

3.2.4 *Data Usage*

The data required to correctly identify a model is a critical aspect when comparing SINDy-PI and implicit-SINDy. Many experimental data sets are limited in volume, and thus our goal is to identify a model with as little data as possible. In this section, we compare the SINDy-PI and implicit-SINDy methods on the challenging yeast glycolysis model [181, 315] given by

$$\dot{x}_1 = c_1 + \frac{c_2 x_1 x_6}{1 + c_3 x_6^4}, \tag{3.13a}$$

$$\dot{x}_2 = \frac{d_1 x_1 x_6}{1 + d_2 x_6^4} + d_3 x_2 - d_4 x_2 x_7, \tag{3.13b}$$

$$\dot{x}_3 = e_1 x_2 + e_2 x_3 + e_3 x_2 x_7 + e_4 x_3 x_6 + f_5 x_4 x_7, \tag{3.13c}$$

$$\dot{x}_4 = f_1 x_3 + e_2 x_4 + f_3 x_5 + f_4 x_3 x_6 + f_5 x_4 x_7, \tag{3.13d}$$

$$\dot{x}_5 = g_1 x_1 + g_2 x_5, \tag{3.13e}$$

$$\dot{x}_6 = h_3 x_3 + h_5 x_6 + h_4 x_3 x_6 + \frac{h_1 x_1 x_6}{1 + h_2 x_6^4}, \tag{3.13f}$$

$$\dot{x}_7 = j_1 x_2 + j_2 x_2 x_7 + j_3 x_4 x_7. \tag{3.13g}$$

Equation (3.13f) is the most challenging equation to discover in this system, and Fig. 3.4 compares the success rate of SINDy-PI and implicit-SINDy in identifying this equation. SINDy-PI uses about 12 times less data than the implicit-SINDy when identifying (3.13f). Details are provided in Appendices A.2 and A.4.

3.2.5 *Comparison for Implicit PDE Identification*

We now investigate the ability of SINDy-PI to discover a PDE with rational terms, given by a modified KdV equation

$$u_t = -u_{xxx} - 6uu_x - \gamma u + \frac{2g_0}{1+u}, \tag{3.14}$$

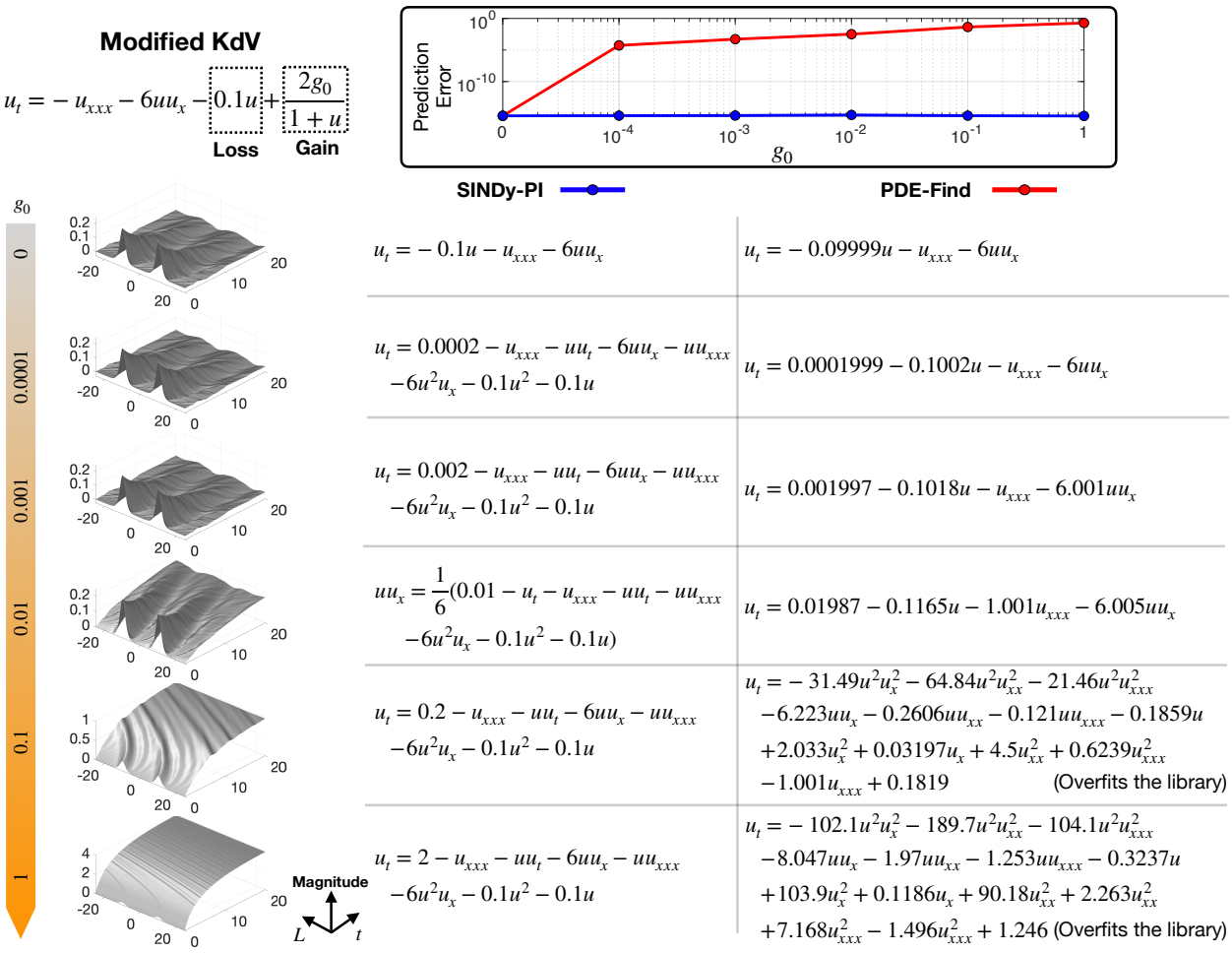


Figure 3.5: Comparison of SINDy-PI and PDE-FIND on an implicit PDE problem given by the modified KdV equation (3.14). As we increase g_0, the rational term begins to play a significant role in the system behavior. For small g_0, PDE-FIND compensates for the effect of the rational term by tuning the other coefficients. When g_0 is large, PDE-FIND overfits the library. SINDy-PI, on the other hand, correctly identifies the rational term.

where γu is a loss term and $2g_0/(1+u)$ is a gain term. We fix $\gamma = 0.1$ and vary the value of g_0 from 0 to 1. As g_0 increases, the implicit term gradually dominates the dynamics. Figure 3.5 shows the results of SINDy-PI and PDE-FIND [12] for different values of g_0. For large g_0, SINDy-PI is able to accurately identify the rational function term, while this is not possible for PDE-FIND, since this term is not in the library. Details of the identification

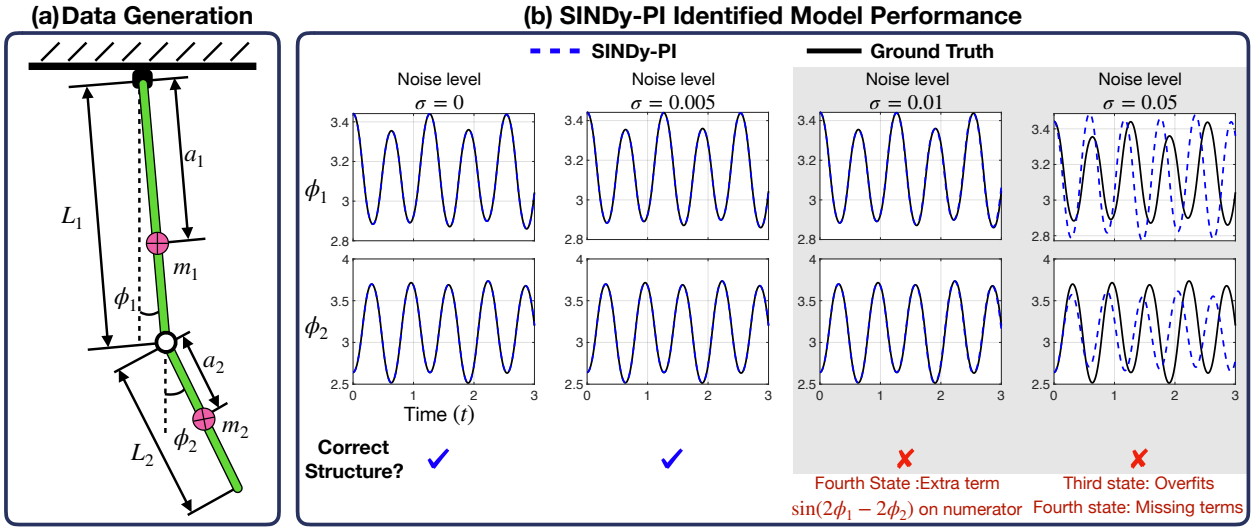


Figure 3.6: Schematic illustration of SINDy-PI identifying a mounted double pendulum system.

process are given in Appendix. A.3.

3.3 Advanced Examples

We will now demonstrate the SINDy-PI framework on several challenging examples, including the double pendulum, an actuated single pendulum on a cart, the Belousov-Zhabotinsky PDE, and the identification of conserved quantities. All examples are characterized by rational nonlinearities, and we were unable to identify them using SINDy or implicit-SINDy, even in the absence of noise.

3.3.1 Mounted Double Pendulum

In our first example, we use SINDy-PI to discover the equations of motion of a mounted double pendulum, shown in Fig. 3.6. The double pendulum is a classic example of chaotic dynamics [7], and was an original challenging example used to demonstrate the capabilities of genetic program for model discovery [14]. Correctly modeling the nonlinear dynamics is vital for accurate control [7].

We simulate the double pendulum dynamics, derived from the Euler-Lagrange equations, and use SINDy-PI to re-discover the dynamics from noisy measurements of the trajectory data. The governing equations and SINDy-PI models are provided in Appendix. A.6. Because these dynamics have rational nonlinearities, the original SINDy algorithm is unable to identify the dynamics, making this a challenging test case. The state vector is given by $\boldsymbol{x} = [\phi_1, \phi_2, \dot{\phi}_1, \dot{\phi}_2]^T$, and the parameters of the simulation are given in Appendix. A.4. The training data is generated from an initial condition $x_{\text{train}} = [\pi + 1.2,\ \pi - 0.6,\ 0,\ 0]^T$, simulated for 10 seconds using a time step of $dt = 0.001$ seconds. The validation data is generated from an initial condition $x_{\text{val}} = [\pi - 1,\ \pi - 0.4,\ 0.3,\ 0.4]^T$, simulated for 3 seconds with time step $dt = 0.001$ seconds.

To test the robustness of SINDy-PI, we add Gaussian noise to both the training and validation data. We test the resulting models using a new testing initial condition $x_{\text{test}} = [\pi + 0.3,\ \pi - 0.5,\ 0,\ 0]^T$. We construct our library $\boldsymbol{\Theta}$ to include over 40 trigonometric and polynomial terms. The most challenging part of this example is building a library with the necessary terms, without it growing too large. The library cannot be too extensive, or else the matrix $\boldsymbol{\Theta}$ becomes ill conditioned, making it sensitive to noise. To reduce the library size, we use one piece of expert knowledge: the trigonometric terms should only consist of ϕ_1 and ϕ_2, the rotational angles of the pendula.

The candidate functions are chosen as a combination of state derivatives and trigonometric functions. Fig. 3.6 shows that SINDy-PI can identify the equations of motion for low noise. For larger noise, SINDy-PI misidentifies the dynamics, although it still has short term prediction ability.

3.3.2 Single Pendulum on a Cart

We now apply SINDy-PI to identify a fractional ODE problem with control input, given by the single pendulum on a cart in Fig. 3.7. SINDy has already been extended to include control inputs [281], although the original formulation doesn't accommodate rational

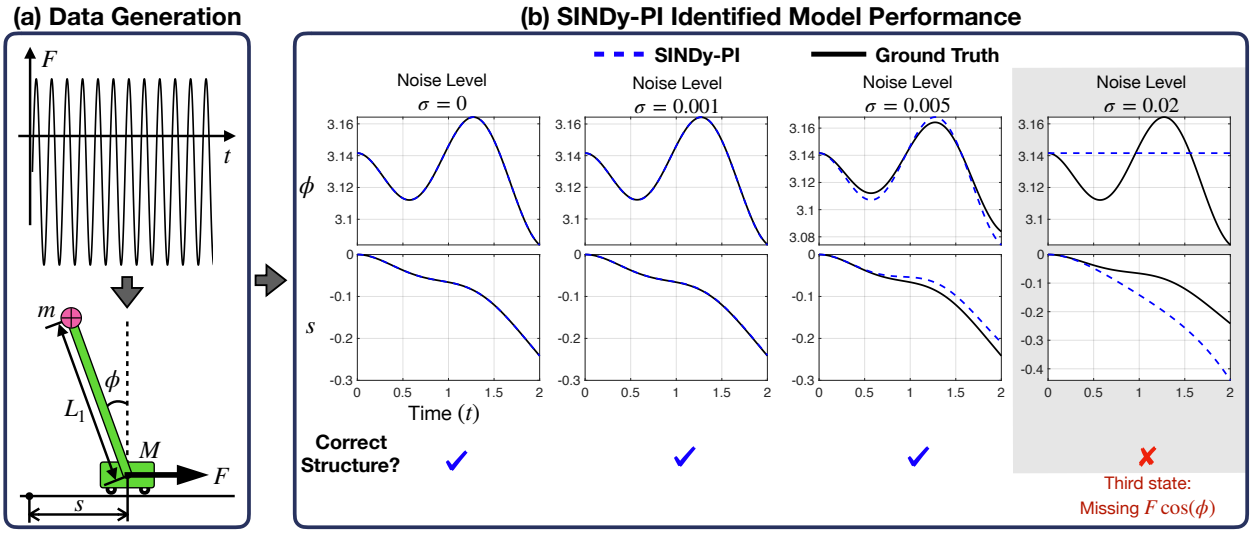


Figure 3.7: SINDy-PI is used to identify the single pendulum on a cart system. Control is applied to the cart, and both the cart and pendulum states are measured. When the measurement noise is small, SINDy-PI can identify the correct structure of the model.

functions.

The dynamics are derived from the Euler-Lagrange equations. All system parameters except for gravity are chosen to be 1, as summarized in Appendix. A.4; the governing equations and SINDy-PI models are shown in Appendix. A.5. The cart position is denoted by s. The state vector is given by $\boldsymbol{x} = [\phi, s, \dot{\phi}, \dot{s}]^T$. The equations of motion are given by

$$\frac{d}{dt}\phi = \dot{\phi}, \tag{3.15a}$$

$$\frac{d}{dt}s = \dot{s}, \tag{3.15b}$$

$$\frac{d}{dt}\dot{\phi} = -\frac{(M+m)g\sin(\phi) + FL_1\cos(\phi) + mL_1^2\sin(\phi)\cos(\phi)\dot{\phi}^2}{L_1^2(M+m-m\cos(\phi)^2)}, \tag{3.15c}$$

$$\frac{d}{dt}\dot{s} = \frac{mL_1^2\sin(\phi)\dot{\phi}^2 + FL_1 + mg\sin(\phi)\cos(\phi)}{L_1(M+m-m\cos(\phi)^2)}, \tag{3.15d}$$

Eq. (3.15) is simulated with a time step of $dt = 0.001$ to generate the training and testing data for model selection. The training data is generated using an initial condition $x_{\text{train}} = [0.3,\ 0,\ 1,\ 0]^T$ with the control input chosen as $F_{\text{train}} = -0.2 + 0.5\sin(6t)$, for time

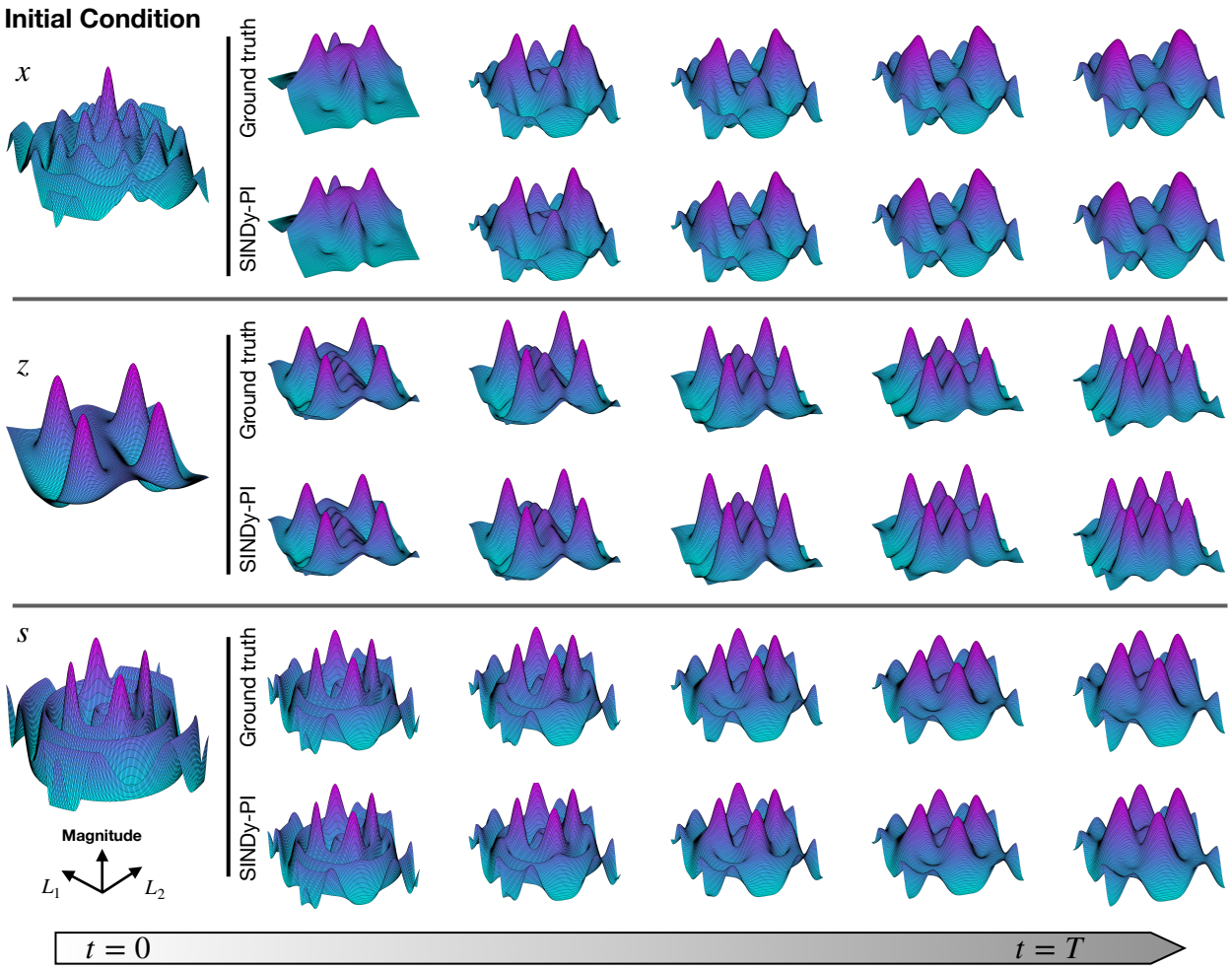


Figure 3.8: SINDy-PI is able to identify the simplified Belousov–Zhabotinsky reaction model.

$t = 0$ to $t = 16$. Similarly, the validation data is generated using an initial condition $x_{\text{val}} = [0.1,\ 0,\ 0.1,\ 0]^T$ with the control input chosen as $F_{\text{val}} = -1 + \sin(t) + 3\sin(2t)$, for time $t = 0$ to $t = 2$.

The library is constructed using a combination of trigonometric and polynomial terms. Around 50 different basis functions are used for the library, and around 10 terms are tested as candidate functions. We add Gaussian noise to all system states. We then test the SINDy-PI model on a testing initial condition $x_{\text{test}} = [\pi,\ 0,\ 0,\ 0]^T$ with control input $F_{\text{test}} = -0.5 + 0.2\sin(t) + 0.3\sin(2t)$ for time $t = 0$ to $t = 2$. Fig. 3.7 shows the

resulting SINDy-PI models. The structure of the model is correctly identified up to a noise magnitude of 0.01. Beyond this noise level, the SINDy-PI identified model only has short term prediction ability.

3.3.3 Simplified Model of the Belousov–Zhabotinsky Reaction

We now apply SINDy-PI to a challenging PDE with rational nonlinearities, a simplified model of the Belousov-Zhabotinsky (BZ) reaction. The simplified BZ reaction model is given by [316]

$$\frac{\partial x}{\partial \tau} = \frac{1}{\varepsilon}\left(\frac{fz(q-x)}{q+x} + x - x^2 - \beta x + s\right) + \frac{D_x}{D_u}\Delta x, \tag{3.16a}$$

$$\frac{\partial z}{\partial \tau} = x - z - \alpha z + \gamma u + \frac{D_z}{D_u}\Delta z, \tag{3.16b}$$

$$\frac{\partial s}{\partial \tau} = \frac{1}{\varepsilon_2}(\beta x - s + \chi u) + \frac{D_s}{D_u}\Delta s, \tag{3.16c}$$

$$\frac{\partial u}{\partial \tau} = \frac{1}{\varepsilon_3}[\alpha z - (\gamma + \frac{\chi}{2})u] + \frac{D_u}{D_u}\Delta u, \tag{3.16d}$$

where x, z, s, and u are dimensionless variables and $\Delta = \frac{\partial^2}{\partial x_s^2} + \frac{\partial^2}{\partial y_s^2}$ denotes the Laplacian operator.

The strong coupling dynamics and implicit behavior in (3.16a) make the data-driven discovery of the simplified BZ reaction challenging when using implicit-SINDy and PDE-FIND. However, SINDy-PI correctly identifies the simplified dynamics of the BZ-Reaction, as shown in Fig. 3.8. To generate the simplified BZ reaction data, we use a spectral method [11, 317] with time horizon $T = 1$ and time step of $dt = 0.001$. We use $n = 128$ discretization points with spatial domain ranging from -10 to 10. The initial condition is chosen to be a mixture of Gaussian functions. 80% of the data is used for training, and the remaining 20% is used for model selection. The right-hand side library is normalized during the sparse regression process. A range of sparsity parameters λ are tested from 0.1 to 1, with increments of 0.1 The other system parameters in (3.16) are given in Appendix. A.4 and the SINDy-PI model is given in Appendix. A.7.

3.3.4 Extracting Physical Laws and Conserved Quantities

In this final example, we demonstrate how to use SINDy-PI to extract governing physical laws and conserved quantities from data. Many systems of interest are governed by Hamiltonian or Lagrangian dynamics. Instead of identifying the ODE or PDE equations of motion, it might be possible to extract the physical laws directly. These equations contain important information about the system and may be more concise, useful, and straightforward than the underlying ODE or PDE. For example, given a Lagrangian, we can derive the equations of motion.

The most difficult aspect of using SINDy-PI to identify a physical law is how to build the library. Conservation laws may contain higher-order derivatives, such as $\ddot{x}$. To include all possible terms, the library may become exceedingly large. The library size will also increase if the system has many states. Large libraries make the sparse regression sensitive to noise. Thus, extracting the physical law from data using SINDy-PI is still challenging due to the lack of constraints when constructing the library function. We only show one example in our paper to demonstrate that it is possible to achieve this using SINDy-PI, but further work is required to reduce the library size so that the sparse regression is robust.

As an example, we consider the double pendulum shown in Fig. 3.9, with the system parameters given in Appendix. A.4. In this case, we also account for the friction in the pendulum joint, with friction constants of $k_1 = 7.2484 \times 10^{-4}$ and $k_2 = 1.6522 \times 10^{-4}$ for the pendulum arms, respectively. In this case, we extract the Lagrangian of the double pendulum [7] using SINDy-PI. To extract this Lagrangian, we simulate the system with initial condition $x_{\text{train}} = [\pi - 0.6,\ \pi - 0.4,\ 0,\ 0]^T$ from $t = 0$ to $t = 15$ with time step $dt = 0.001$. The resulting model is shown in Fig. 3.9.

3.4 Conclusions and Future Work

In this paper, we develop SINDy-PI (parallel,implicit), a robust variant of the SINDy algorithm to identify implicit dynamics and rational nonlinearities. SINDy-PI overcomes

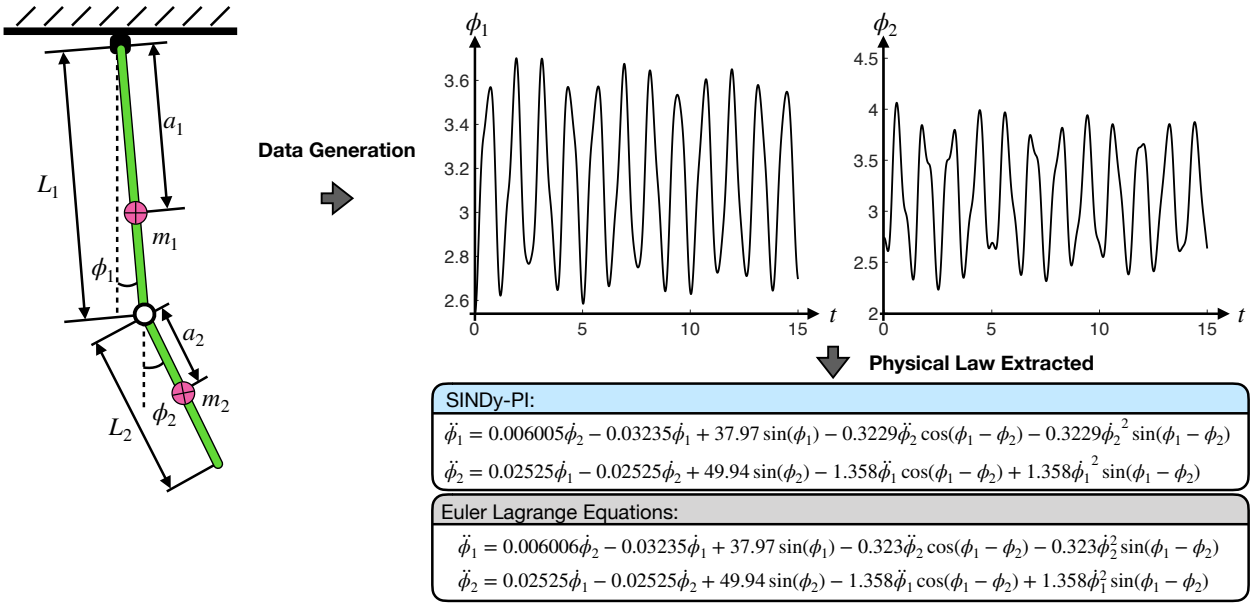


Figure 3.9: SINDy-PI is used to extract the conserved quantity for a double pendulum.

the sensitivity of the previous implicit-SINDy approach, which is based on a null-space calculation, making it highly sensitive to noise. Instead, we introduce both parallel and constrained optimizations to test candidate terms in the dynamics, making the new SINDy-PI algorithm as robust as the original SINDy algorithm. We also extend the algorithm to incorporate external forcing and actuation, making it more applicable to real-world systems. We demonstrate this approach on several challenging systems with implicit and rational dynamics, including ODEs, actuated systems, and PDEs. In particular, we discover the implicit dynamics for a simplified model for the BZ chemical reaction PDE, the double pendulum mechanical system, and the yeast glycolisis model, which have all been challenging test cases for advanced identification techniques. Throughout these examples, we demonstrate considerable noise robustness and reductions to the data required, over the previous implicit-SINDy algorithm.

Despite the advances outlined here, there are still many important avenues of future work. One limitation of this approach, and of SINDy in general, is in the design of the library of candidate functions. The goal is a descriptive library, but the library size

grows rapidly, which in turn makes the sparse regression ill-conditioned; other issues effecting robustness are discussed in Appendix. A.9. Recently, tensor approaches have been introduced to alleviate this issue, making libraries both descriptive and tractable [298], and this is a promising approach that may be incorporated in SINDy-PI as well. More generally, automatic library generation, guided by expert knowledge, is an important topic. Other research directions will involve parameterizing elements of the library, so that the algorithm simultaneously identifies the model structure and the parameters of the sparsely selected terms. Recent unified optimization frameworks, such as SR3 [215, 216], may make this possible. Model selection is another key area that will required focused attention. Balancing accuracy on test data, sparsity of the model, and the potential for overfitting are all serious concerns. The sparse regression and optimization may also be improved for better noise robustness. Finally, modifying SINDy-PI to incorporate prior physical knowledge and to only model the discrepancy with an existing model [178] will be the focus of ongoing work.

Chapter 4

AUTOMATIC DIFFERENTIATION TO SIMULTANEOUSLY IDENTIFY NONLINEAR DYNAMICS AND EXTRACT NOISE PROBABILITY DISTRIBUTIONS FROM DATA

4.1 Introduction

As with all system identification algorithms, noisy measurements compromise the accuracy and robustness of the model discovery procedure. Moreover, many optimization frameworks rely explicitly on the assumption of Gaussian noise, which is rarely true in the real world. Recently, Rudy et al. [318] developed a novel optimization framework for separating signal and noise from noisy time-series data by identifying a deep NN model for the signal from numerical time-stepping constraints such as a Runge-Kutta. In this Chapter, we build on this framework and leverage automatic differentiation [319] in the optimization procedure to simultaneously denoise data and identify sparse nonlinear models via SINDy. This new architecture yields significant improvements in model discovery, including superior separation of the signal from noise while simultaneously characterizing the noise distribution.

SINDy has emerged as a flexible and promising architecture for model discovery due to its inherent parsimonious representation of dynamics. The SINDy framework relies on sparse regression on a library of candidate model terms to select the fewest terms required to describe the observed dynamics [180]. Specifically, SINDy is formulated as an over-determined linear system of equations $\mathbf{A}\boldsymbol{\xi} = \mathbf{b}$, where A is a library matrix, b represents the measurement data, and ξ represents the sparse selection vector, and the sparsity of the solution is promoted by the ℓ_0-norm $\|\boldsymbol{\xi}\|_0$. Thus sparsity is a proxy for parsimony, interpretability, and generalizability. Measurement noise, however, is always

present, and it corrupts the ability of the SINDy regression framework, and indeed any other model discovery paradigm, to accurately extract governing models.

There are many variants of sparse regression, all of which typically attempt to approximate the solution to an NP-hard, ℓ_0-norm penalized regression. Sparsity-promoting methods like the LASSO [186, 187] use the ℓ_1-norm as a proxy for sparsity since tractable computations can be performed. The iterative least-squares thresholding technique of the SINDy algorithm promotes sparsity through a sequential procedure. Recently, Zhang and Schaeffer [184] have provided several rigorous theoretical guarantees on the convergence of the SINDy algorithm. Specifically, they proved that the algorithm approximates local minimizers of an unconstrained ℓ_0-penalized least-squares problem, which allows them to provide sufficient conditions for general convergence, the rate of convergence, and conditions for one-step recovery. Using a relaxed formulation, Champion et al. [215] show how the SINDy regression framework can accommodate additional structure, robustness to outliers, and nonlinear parameter estimation using the *sparse relaxed regularized regression* (SR3) formulation [216]. SINDy results in interpretable models, and it has been widely applied in many scientific disciplines [178, 194, 209, 254, 277, 287, 297, 300, 303, 304, 320–325]. Moreover, it has been extended to various applications [12, 177, 178, 181, 200, 200, 209, 211, 215, 222, 233, 234, 236, 241, 251, 254, 267, 280, 281, 297, 298, 302, 305, 309, 326–328].

Despite its flexibility, modularity, and extensibility, SINDy and its variants typically rely on approximating time-derivative of the measured time-series data. Computing derivatives of noisy measurement data is known to be a challenging problem, with many algorithmic innovations and mathematical architectures developed to produce accurate derivative approximations [329]. These methods include finite-differences, spectral methods [11], spline smoothing, filtering procedures, polynomial fitting, low-rank projection, and total variations, to highlight some of the diverse techniques employed for this critical task of scientific computing. This task is made even more difficult, depending upon the noise statistics. Gaussian noise is often easier to learn and characterize than noise distributions that have non-zero means and are not symmetric. Ultimately, there is a need for

methods that are robust to noisy measurements and diverse probability distributions.

Recent innovations in automatic differentiation have enabled the solution of an optimization problem directly related to the computation of the required derivatives [318]. Since its inception, automatic differentiation has been widely used in the machine learning community to enable complicated optimization problems without manually computing Jacobians [120, 318, 319, 330–336]. More recently, this approach has been used with NNs to separate a signal from noise and model the signal when a model is unknown [318], and to improve Kalman smoothing when the governing equations are known [333]. The success of these algorithms suggest that they could be leveraged for noise signal separation in the SINDy framework. In this work, we extend this simultaneous de-noising and discovery approach to SINDy. Specifically, automatic differentiation enables differentiation with respect to the functions in the SINDy library, thus circumventing a direct differentiation of the noisy time-series data. The modified SINDy algorithm is more robust to noise and further allows for an explicit characterization (discovery) of the underlying probability distribution of the noise, something that current state-of-the-art methods cannot do and is a unique feature of our method.

In Sec. 4.2, we illustrate the modified SINDy algorithm. In Sec. 4.3, we show the comparison between modified SINDy and noise signal separation approach based on the NN proposed by Rudy et al. [318]. In Sec. 4.4, we show the use of modified SINDy on various numerical examples and show how modified SINDy can be used to identify noise distributions. In Sec. 4.5, we show our conclusions and possible future improvements.

4.2 Methods

In what follows, we introduce the basic mathematical architecture behind the SINDy algorithm, demonstrating explicitly its sensitivity to noisy measurements. This guides our introduction of the modified SINDy for simultaneously learning the system model and denoising the signal.

4.2.1 Sparse Identification of Nonlinear Dynamics

The SINDy algorithm [180] provides a principled, data-driven discovery method for nonlinear dynamics of the form shown in Sec. 2.1 given data matrix $\mathbf{X}$, derivative matrix $\dot{\mathbf{X}}$ and candidate library $\boldsymbol{\Theta}(\mathbf{X})$. In practice, noise-free measurements of $\mathbf{x}(t)$ are not available, and only the full state noisy measurement

$$\mathbf{y}(t) = \mathbf{x}(t) + \mathbf{n}(t), \tag{4.1}$$

is provided to SINDy from sensors, where $\mathbf{y}(t) = [y_1(t), y_2(t), \cdots, y_n(t)] \in \mathbb{R}^{1\times n}$ is noisy measurement and $\mathbf{n}(t) = [n_1(t), n_2(t), \cdots, n_n(t)] \in \mathbb{R}^{1\times n}$ is the noise added to true state. Thus, Eq. (2.4) then becomes

$$\dot{\mathbf{Y}} = \dot{\mathbf{X}} + \dot{\mathbf{N}} = \boldsymbol{\Theta}(\mathbf{Y})\boldsymbol{\Xi} = \boldsymbol{\Theta}(\mathbf{X} + \mathbf{N})\boldsymbol{\Xi}, \tag{4.2}$$

where $\mathbf{Y} = [\mathbf{y}(t_1); \mathbf{y}(t_2); \cdots; \mathbf{y}(t_m)] \in \mathbb{R}^{m\times n}$ is noisy measurement matrix formed by m row vectors measurement of size $1 \times n$ and $\mathbf{N} = [\mathbf{n}(t_1); \mathbf{n}(t_2); \cdots; \mathbf{n}(t_m)] \in \mathbb{R}^{m\times n}$ is noise matrix also formed by m row vector of size $1 \times n$. From Eq. (4.2), note that the solution $\boldsymbol{\Xi}$ is no longer the same $\boldsymbol{\Xi}$ shown in Eq. (2.4) due to the presence of noise. Moreover, the noise will be magnified when approximating the derivatives $\dot{\mathbf{X}}$ by a factor of $\mathcal{O}(1/dt)$ [12], and it will non-linearly corrupt the library matrix $\boldsymbol{\Theta}$. Extensive research has been done to improve the robustness of the SINDy framework. The integral formulation [236] and weak formulation [232–234, 309] improved the regression robustness by avoiding taking derivative of noisy data. Other approaches, such as subsampling [229], increased the noise robustness of the SINDy framework by doing regression on the subsampled measurement that has less noise. Corrupt data can also be handled with methods from robust statistics [215, 251]. In the next section, we introduce an alternative approach that simultaneously learns the noise $\mathbf{N}$ while using the denoised data to perform model identification.

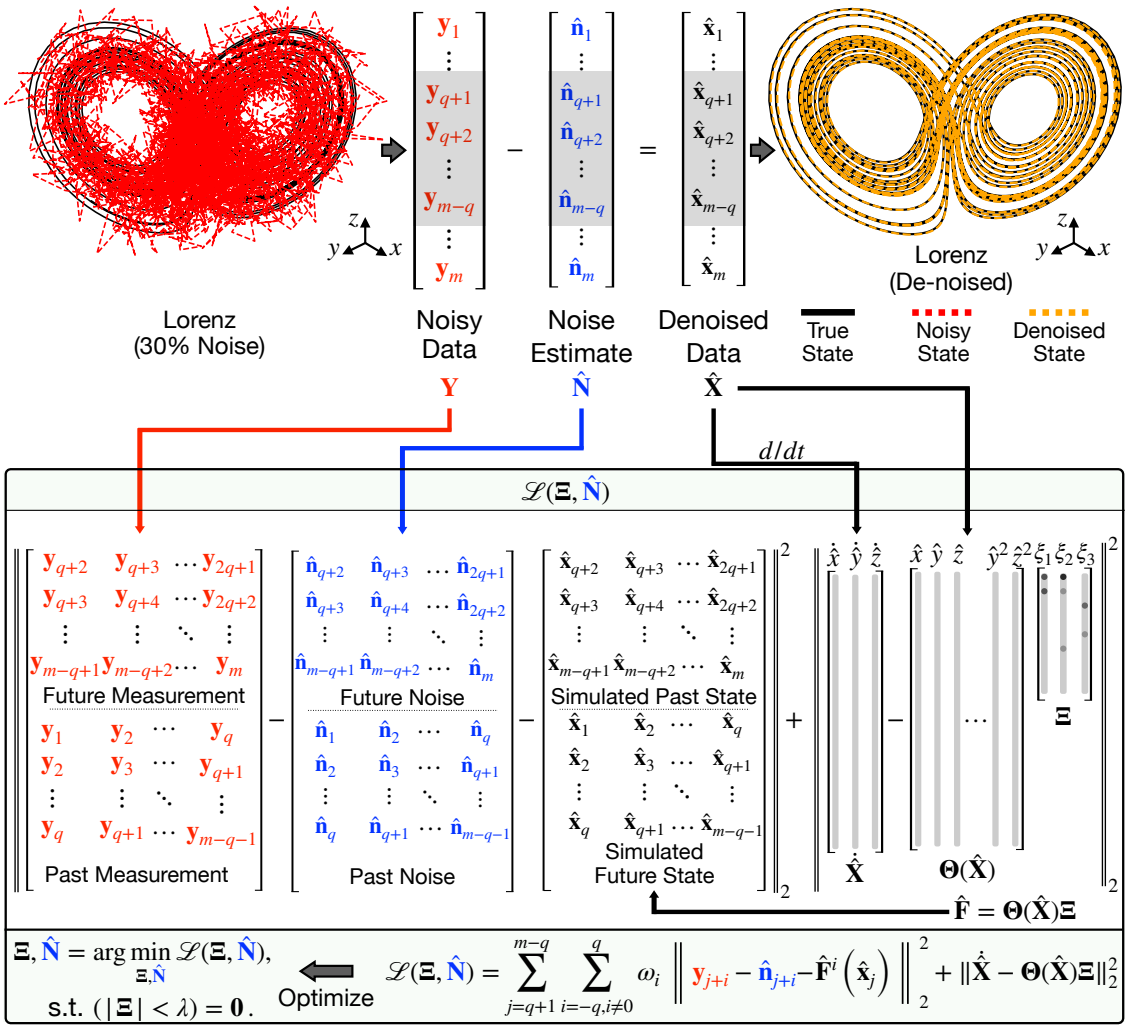


Figure 4.1: This figure illustrates modified SINDy algorithm. The goal is to learn the system model $\boldsymbol{\Theta}(\mathbf{X})\boldsymbol{\Xi}$ and noise $\mathbf{N}$. The noise is subtracted from the measurement to obtain the clean data. To achieve this, the estimated noise $\hat{\mathbf{N}}$ is set as an optimization parameter and the cost function $\mathcal{L}(\boldsymbol{\Xi}, \hat{\mathbf{N}})$ is minimized. This optimization is performed 8 times, and the small values of $|\boldsymbol{\Xi}|$ is enforced to be zero for the remainder of the optimization process. The proposed de-noising algorithm is related to the scheme of Rudy et al. [318], but embedded into the SINDy model discovery framework.

4.2.2 *Simultaneously Denoising and Learning System Model*

To improve the noise robustness of the SINDy regression, we determine the estimated noise $\hat{\mathbf{n}}(t) \in \mathbb{R}^{1\times n}$ as a hyper-parameter and formulate $\hat{\mathbf{N}} = [\hat{\mathbf{n}}(t_1); \hat{\mathbf{n}}(t_2); \cdots; \hat{\mathbf{n}}(t_m)] \in \mathbb{R}^{m\times n}$

in order to optimize the difference between the estimated derivative and system's vector field such that

$$e_d = \|\dot{\hat{\mathbf{X}}} - \boldsymbol{\Theta}(\hat{\mathbf{X}})\boldsymbol{\Xi}\|_2^2, \tag{4.3}$$

where $\hat{\mathbf{X}} = \mathbf{Y} - \hat{\mathbf{N}}$ is formed by m estimated true states $\hat{\mathbf{x}}(t) \in \mathbb{R}^{1\times n}$, and e_d is the derivative approximation error. For details on calculating this error, please see Appendix. B.12. Note that $\hat{\mathbf{X}} = [\hat{\mathbf{x}}(t_1); \hat{\mathbf{x}}(t_2); \cdots; \hat{\mathbf{x}}(t_m)] \in \mathbb{R}^{m\times n}$. When $\hat{\mathbf{N}} = \mathbf{N}$, the effect of noise can be eliminated, and the accuracy of SINDy will be significantly improved. However, there exist many trivial solutions for minimizing the Eq. (4.3) with two uncorrelated optimization parameters $\hat{\mathbf{N}}$ and $\boldsymbol{\Xi}$. Thus, an additional constraint is needed to regularize Eq. (4.3).

The additional constraint proposed here uses the estimated vector field of the system model similar to the one proposed by Rudy et al. [318]. Equation (2.5) gives the estimate $\boldsymbol{\Theta}(\mathbf{x}(t))\boldsymbol{\Xi}$ of the true vector field $\boldsymbol{f}(\mathbf{x}(t))$. Integrating over a segment of time t_j to t_{j+1} gives the integrated vector field, or flow map,

$$\mathbf{x}(j+1) = \mathbf{F}(\mathbf{x}(j)) = \mathbf{x}(j) + \int_{t_j}^{t_{j+1}} \boldsymbol{\Theta}(\mathbf{x}(\tau))\boldsymbol{\Xi}\, d\tau. \tag{4.4}$$

This can be generalized to integrate the system either forward or backward in time q steps. This gives

$$\mathbf{x}(j+q) = \mathbf{F}^q(\mathbf{x}(j)) = \mathbf{x}(j) + \int_{t_j}^{t_{j+q}} \boldsymbol{\Theta}(\mathbf{x}(\tau))\boldsymbol{\Xi}\, d\tau. \tag{4.5}$$

To obtain the $\mathbf{x}(j+q)$ in Eq. (4.5), a numerical simulation scheme such as Runge–Kutta can be used. In what follows, we employ a 4th-order Runge-Kutta method to simulate the dynamics forward/backward in time q-steps. Similar to Eq. (4.5), when the noisy measurement data $\mathbf{y}$ is given, the estimated state $\hat{\mathbf{x}} = \mathbf{y} - \hat{\mathbf{n}}$ satisfies

$$\mathbf{y}(j+q) - \hat{\mathbf{n}}(j+q) = \hat{\mathbf{x}}(j+q) = \hat{\mathbf{F}}^q(\hat{\mathbf{x}}(j)) = \hat{\mathbf{x}}(j) + \int_{t_j}^{t_{j+q}} \boldsymbol{\Theta}(\hat{\mathbf{x}}(\tau))\boldsymbol{\Xi}\, d\tau, \tag{4.6}$$

when $\hat{\mathbf{n}} = \mathbf{n}$ and the exact value of $\boldsymbol{\Xi}$ is known. Thus, by minimizing

$$e_{s,j} = \sum_{i=-q, i\neq 0}^{q} \omega_i \|\mathbf{y}(j+i) - \hat{\mathbf{n}}(j+i) - \hat{\mathbf{F}}^i(\hat{\mathbf{x}}(j))\|_2^2, \tag{4.7}$$

the optimization parameters $\hat{\mathbf{N}}$ and $\boldsymbol{\Xi}$ are coupled, resulting in additional structural constraint of the model. The parameter ω_i is used to account for the numerical error and is set to $\omega_i = c^{|i|-1}$, where $0 < c \leq 1$ is a constant (throughout this paper, we use $c = 0.9$). The use of ω suggests that the simulation error too far ahead in the future, or too far backward in the past, should be penalized less due to the error of the numerical simulation scheme. The error incurred by simulating the vector filed forward/backward on the entire trajectory can be written as

$$e_s = \sum_{j=q+1}^{m-q} e_{s,j} = \sum_{j=q+1}^{m-q} \sum_{i=-q, i\neq 0}^{q} \omega_i \|\mathbf{y}(j+i) - \hat{\mathbf{n}}(j+i) - \hat{\mathbf{F}}^i(\hat{\mathbf{x}}(j))\|_2^2. \tag{4.8}$$

Using subscripts to represent the time step, the final cost function is then

$$\mathcal{L}(\boldsymbol{\Xi}, \hat{\mathbf{N}}) = e_s + e_d = \sum_{j=q+1}^{m-q} \sum_{i=-q, i\neq 0}^{q} \omega_i \|\mathbf{y}_{j+i} - \hat{\mathbf{n}}_{j+i} - \hat{\mathbf{F}}^i\left(\hat{\mathbf{x}}_j\right)\|_2^2 + \|\dot{\hat{\mathbf{X}}} - \boldsymbol{\Theta}(\hat{\mathbf{X}})\boldsymbol{\Xi}\|_2^2, \tag{4.9}$$

which is the summation of the derivative approximation error e_d and simulation error e_s. The optimization problem to simultaneously denoise and learn the system model can then be written as

$$\begin{aligned} \boldsymbol{\Xi}, \hat{\mathbf{N}} &= \arg\min_{\boldsymbol{\Xi}, \hat{\mathbf{N}}} \mathcal{L}(\boldsymbol{\Xi}, \hat{\mathbf{N}}), \\ \text{s.t.} \quad & (|\boldsymbol{\Xi}| < \lambda) = \mathbf{0}. \end{aligned} \tag{4.10}$$

The global optimal solution for Eq. (4.10) needs to satisfy $\hat{\mathbf{N}} = \mathbf{N}$ and $\boldsymbol{f}(\mathbf{x}) = \boldsymbol{\Theta}(\mathbf{x})\boldsymbol{\Xi}$. However, the global optimum of Eq. (4.9) is not unique, as Appendix B.11 suggests. To solve for Eq. (4.10), it is necessary to calculate the Jacobian $\partial\mathcal{L}/\partial\hat{\mathbf{N}}$ and $\partial\mathcal{L}/\partial\boldsymbol{\Xi}$, which is a difficult task to do analytically or computationally. However, recent automatic dif-

ferentiation packages such as Tensorflow [330] and Julia Flux [337] make it possible to directly extract the gradients of $\mathcal{L}$ with respect to $\hat{\mathbf{N}}$ and $\boldsymbol{\Xi}$. Other alternatives include JAX MD [338, 339]. This allows us to solve the optimization problem in Eq. (4.10) easily using gradient descent method such as Adam [340]. Throughout this paper, we use the Tensorflow 2.0 and Adam optimizer to solve the Eq. (4.10). Moreover, to enforce the sparsity of the identified model, a thresholding approach [180] is used and the Eq. (4.10) is solved for N_{loop} times (the sparsity is enforced to the model structure $\boldsymbol{\Xi}$ not the noise $\mathbf{N}$). Each iteration uses the previous iteration's optimization result $\hat{\mathbf{N}}$ as the initial guess of the new iteration. The values of $\hat{\mathbf{N}}$ is also used to calculate the estimated state $\hat{\mathbf{X}}$, which is used to calculate the new estimated values of the selection parameter $\boldsymbol{\Xi}$. Furthermore, if the elements in $|\boldsymbol{\Xi}|$ are smaller than a threshold λ at the end of an optimization loop, those elements will be constrained to zero for the remainder of the optimization process. Fig. 4.1 illustrates this process, and Appendix. B.1 shows the detailed algorithm for simultaneous denoising and sparse model identification. Some guidance on the selection of the hyper-parameters λ, q, and N_{loop} is given in Appendices. B.2, B.3, and B.4.

4.3 Performance Comparison with Neural Network Denoising Approach

The advocated optimization framework of modified SINDy is compared with a NN denosing approach by Rudy et al. [318]. Additionally, the robustness to noise and the amount of data is considered.

4.3.1 Performance Criteria

For ease of comparison, we use the same performance criteria developed by Rudy et al. [318]. Specifically, these are the vector field error $E_{\boldsymbol{f}}$, the noise identification error $E_{\mathbf{N}}$, and the prediction error $E_{\mathbf{F}}$. The vector filed error is

$$E_{\boldsymbol{f}} = \frac{\sum_{i=1}^{m} \left\| \boldsymbol{f}\left(\mathbf{x}_i\right) - \hat{\boldsymbol{f}}\left(\mathbf{x}_i\right) \right\|_2^2}{\sum_{i=1}^{m} \left\| \boldsymbol{f}\left(\mathbf{x}_i\right) \right\|_2^2}, \tag{4.11}$$

which calculates the relative squared ℓ_2 error between the true vector filed and identified vector field $\hat{\boldsymbol{f}}$. The noise identification error is

$$E_{\mathbf{N}} = \frac{1}{m}\sum_{i=1}^{m} \|\mathbf{n}_i - \hat{\mathbf{n}}_i\|_2^2 , \tag{4.12}$$

which is the mean ℓ_2 difference between the true noise $\mathbf{N}$ and identified noise $\hat{\mathbf{N}}$. The prediction error is

$$E_{\mathbf{F}} = \frac{1}{\|\mathbf{X}\|_F^2}\sum_{i=1}^{m-1} \left\|\mathbf{x}_i - \hat{\mathbf{F}}^i\left(\mathbf{x}_1\right)\right\|_2^2 , \tag{4.13}$$

and it calculates the difference between forward simulation trajectory and true trajectory. For comparison of modified SINDy and recently published Weak-SINDy [233], as shown in Appendix. B.6, two more performance criteria are used. The first one is the normalized parameter error

$$E_{\boldsymbol{p}} = \frac{\|\boldsymbol{\Xi} - \hat{\boldsymbol{\Xi}}\|_2}{\|\boldsymbol{\Xi}\|_2}, \tag{4.14}$$

which reflects how much the identified parameters $\hat{\boldsymbol{\Xi}}$ is off from the true parameters $\boldsymbol{\Xi}$. The other one is the success rate, which describes the percentage of identifying the model's correct structure in multiple trials.

4.3.2 Robustness to Noise

The Lorenz attractor is used as an example to test the noise robustness of the the approach. The model of the chaotic Lorenz is

$$\begin{aligned} \dot{x} &= \sigma(y - x), \\ \dot{y} &= x(\rho - z) - y, \\ \dot{z} &= xy - \beta z, \end{aligned} \tag{4.15}$$

where $\sigma = 10$, $\rho = 28$, and $\beta = 8/3$. The Lorenz attractor is simulated with initial condition $x_0 = [5, 5, 25]$, $T = 25$, and $dt = 0.01$. The prediction step is chosen as $q = 3$ for both

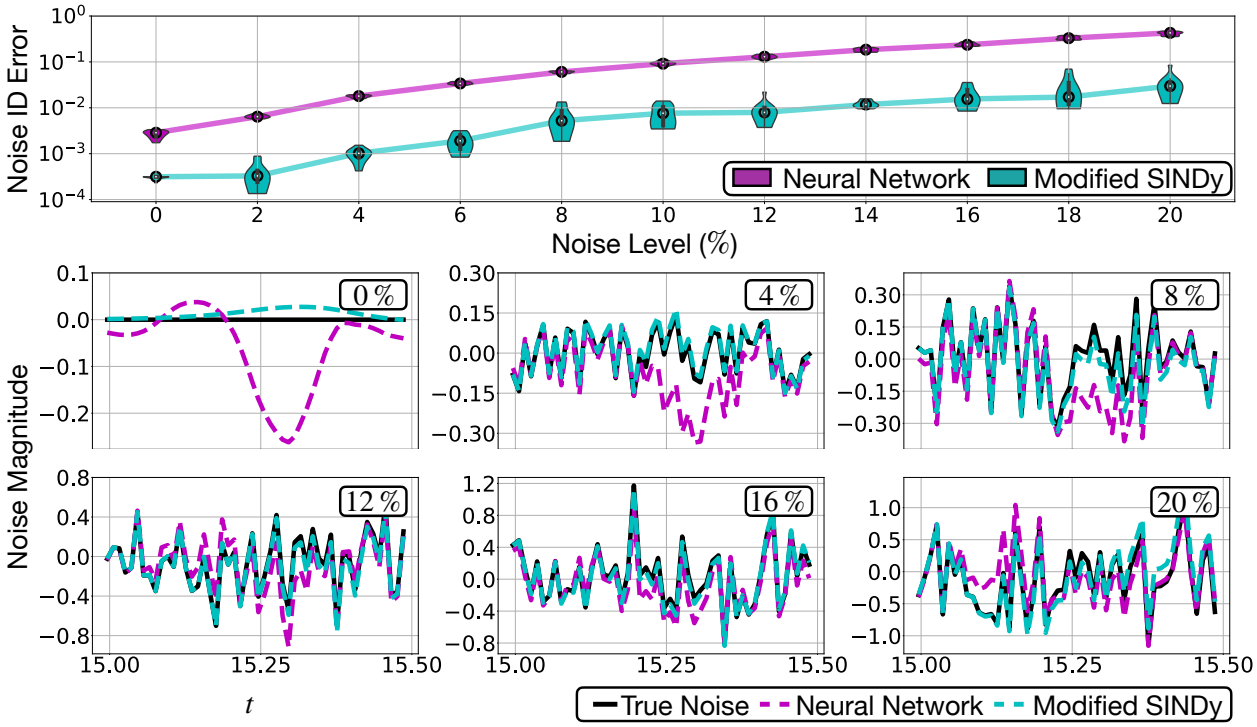


Figure 4.2: Top: Noise identification error of modified SINDy (labeled as SINDy) and NN denoising approach by Rudy et al. [318]. The black circle represents the median of ten runs while the violin shape represents the distribution of error. The modified SINDy approach shows better noise identification error. Bottom: Comparison between the average noise applied to the Lorenz system and the noise identified by the two approaches. As shown on the left, both approaches can not produce the correct zero noise result when no noise is applied, which happens since there is a tiny difference between the learned dynamics and true dynamics.

approaches compared and $N_{loop} = 6$ for our proposed method. Unless otherwise noted, Adam optimizer is used to optimize the problem with maximum iteration set to 5000 for modified SINDy and 30000 for NN approach [318]. Different magnitudes of Gaussian noise are added to generate the noisy training data. The noise level is defined as

$$\text{Noise Level } (\%) = \sqrt{\frac{\text{var(Noise)}}{\text{var(Signal)}}} \times 100\% = \frac{\text{std(Noise)}}{\text{std(Signal)}} \times 100\%. \tag{4.16}$$

For each noise level, 10 different sets of noisy data are generated and used as data for both approaches. The NN approach [318] uses the same set up as [318], with 3 hidden layers, and each layer having 64 neurons. Moreover, the regularization parameter is chosen as 10^{-8}, and the penalty for $\hat{\mathbf{N}}$ is chosen as 10^{-5}. Unless otherwise noted, we use the same set up for all the NNs in this paper. For modified SINDy, the library is constructed with terms up to second order (not including the constant term). Moreover, the value of the sparsity parameter λ varies based on the noise added. For most of the case, $\lambda = 0.1$. A Tikhonov regularization approach is used to pre-smooth the noisy data as in [318], although we have found that pre-smoothing does not affect the results appreciably when using zero-mean noise.

Fig. 4.2 show the noise identification error of the NN approach [318] and the modified SINDy approach. The vector field error and short term prediction error can be seen in Fig. 4.3. For all the noise levels, modified SINDy correctly identified the Lorenz model. To calculate the prediction error, the identified model is simulated 6 seconds forward in time, with $dt = 0.01$, for both modified SINDy and NN denoising approach [318]. Fig. 4.3 suggests that modified SINDy identified model has better performance when simulated forward in time. Other useful way to determine the forward simulation accuracy is by using the Lyapunov exponent of the model. However, unlike the standard way of defining the Lyapunov exponent [341], where the parameters of the model are perfect and only initial conditions of the simulation is perturbed by ϵ, in the situation shown in Fig. 4.3, we have the exact opposite. In Fig. 4.3, the parameters of the identified system is off by ϵ while the initial conditions are perfectly known. Thus, some modification of the definition of Lyapunov exponent is needed before it can be used to define the model forward simulation accuracy. Appendix. B.5 shows noise robustness comparison between the modified SINDy and original SINDy [180]. In general, the modified SINDy is about 2 times more robust than orignal SINDy [180]. A comparison between modified SINDy and the recently developed Weak-SINDy approach [233] is presented in Appendix. B.6.

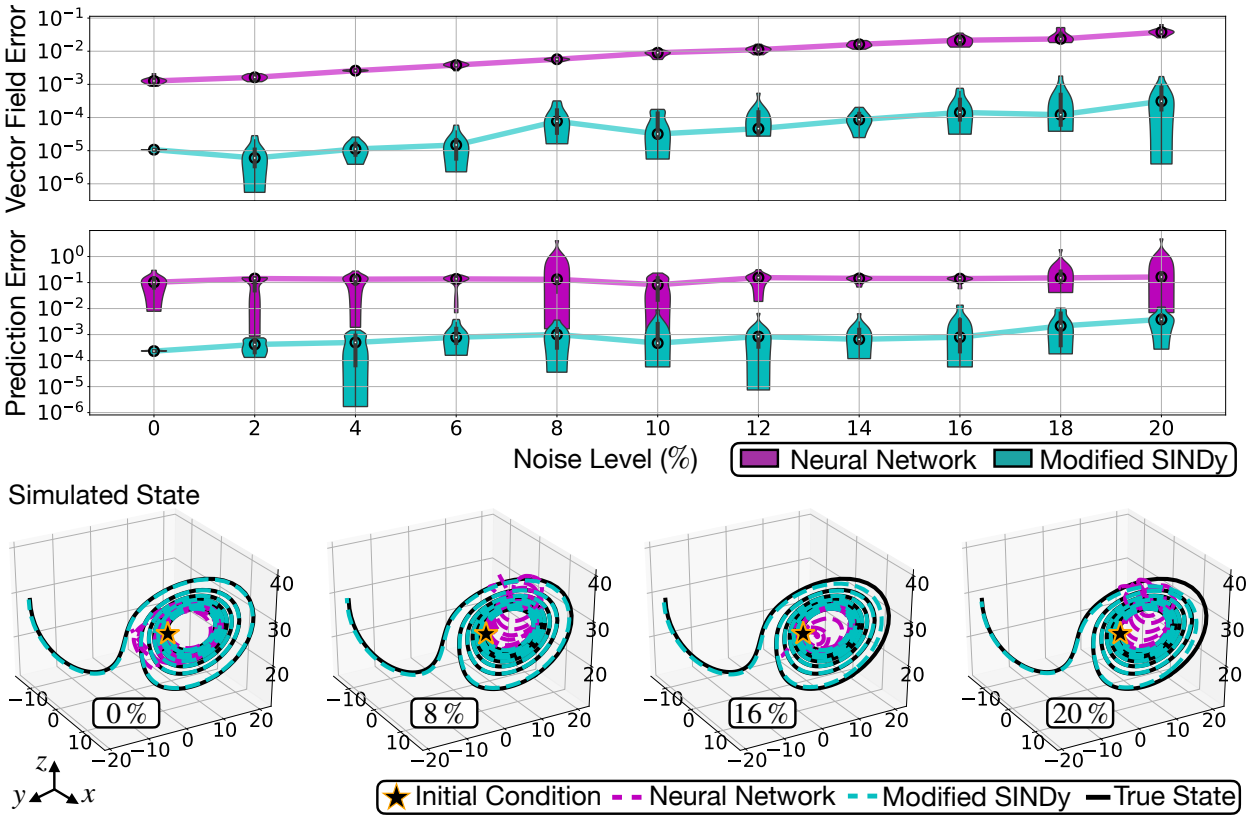


Figure 4.3: Top: The vector field error and prediction error of modified SINDy (labeled as SINDy) and NN denoising approach by Rudy et al. [318] is shown. The black dot is the median of the 10 runs, and the violin shape represents the distribution of the error. Bottom: The simulated trajectory is shown with the initial condition chosen as $x_0 = [5, 5, 25]$.

4.3.3 Robustness to Data Length

We also compare the performance of the NN denoising approach by Rudy et al. [318] and modified SINDy under different data usage with a fixed noise level. The minimum amount of data needed by modified SINDy to correctly identify the system model is shown by using Lorenz attractor as an example. To perform the numerical experiment, the same initial point, $x_0 = [-5, 5, 25]$, is used to generate noise-free data of different temporal lengths. The time step is fixed at $dt = 0.01$ with 10% of Gaussian noise added to generate noisy training data. The success rate of modified SINDy is calculated to indicate

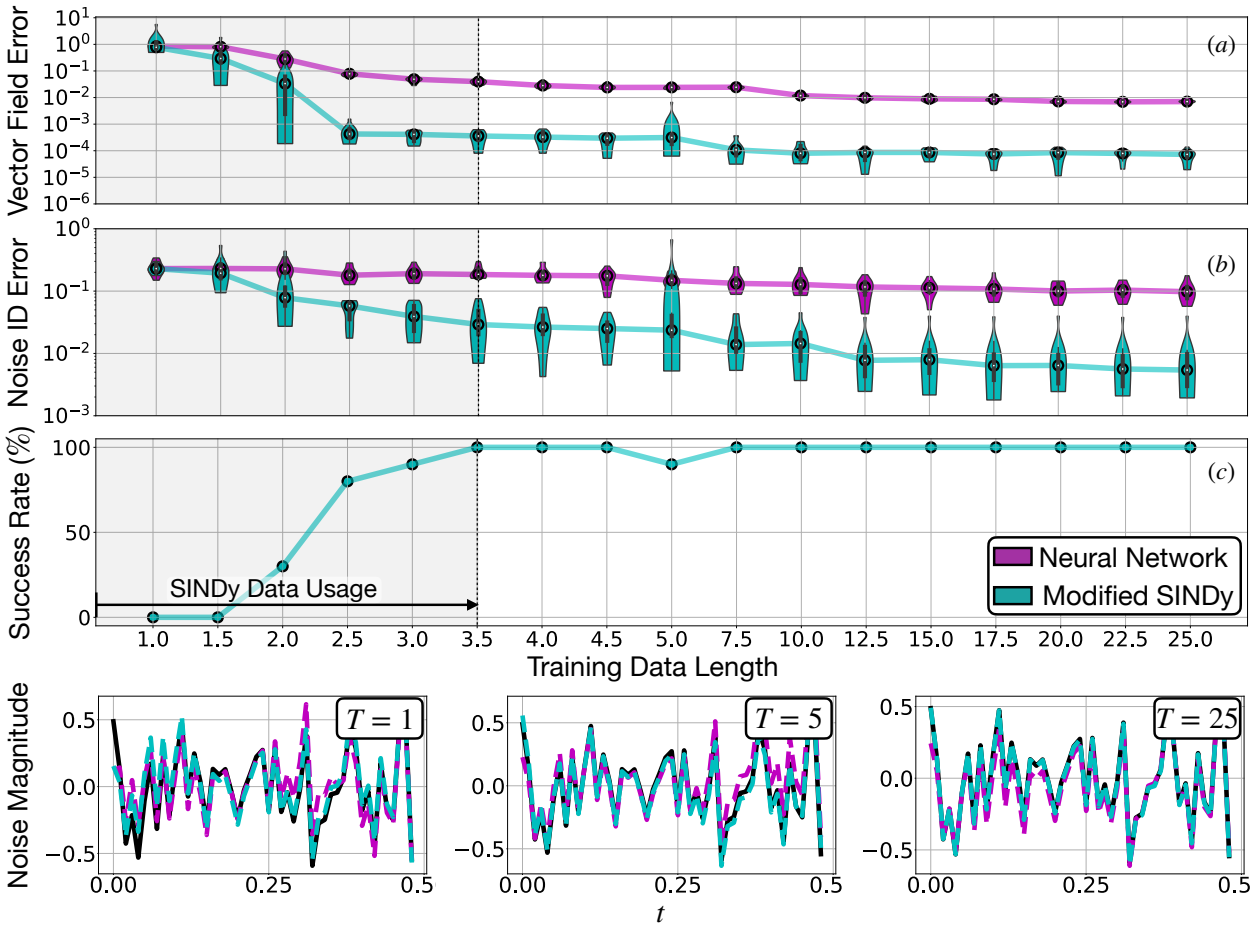


Figure 4.4: (a), (b): As the training data length increases, the vector field error and noise identification error of modified SINDy (labeled as SINDy) and NN denoising approach by Rudy et al. [318] decreases. (c) The modified SINDy can use 3.5 seconds of data to identify the system model with 100% accuracy. There is a tiny drop in the success rate when the training data length is 5 seconds due to the choice of a fixed thresholding parameter. By using a larger thresholding parameter, the success rate can be back to 100%. Moreover, since the NN is a black box model, we won't be able to determine whether it learned the correct symbolic model or not, thus success rate of NN is not plotted. Bottom: A comparison of averaged true noise and identified noise by two approaches is shown.

the minimum amount of data needed to identify the correct system model. The prediction error is not shown since the simulation of the identified model in the low data limit is not stable. With a learning rate of 0.001, Adam is used to optimize the problem with

the prediction step set to $q = 3$ for both approaches. A fixed thresholding parameter $\lambda = 0.1$ with $N_{loop} = 6$ is used for modified SINDy and the library is constructed with up to second order terms (without constant term added). Fig. 4.4 suggests that when the correct parameters and library is used for modified SINDy, it will out-perform the NN denoising approach by Rudy et al. given the same amount of data.

4.4 Results

In this section, we demonstrate the ability of modified SINDy to separate signal and noise while learning the system model. The Van der Pol oscillator will be used as the example test case to show that modified SINDy can identify the correct distribution of the Gaussian noise added to the system. Additionally, we highlight several other examples tested with modified SINDy and summarize the performance. Furthermore, as a more advanced example, we show that modified SINDy can be used to separate non-Gaussian, non-zero mean, and non-symmetric noise distributions from the dynamics. Finally, we show how modified SINDy can be integrated to the discrepancy modeling approach [178].

4.4.1 Van der Pol Oscillator

The Van der Pol oscillator is used as our test case to demonstrate the ability of modified SINDy to denoise and learn the system dynamics simultaneously. The Van der Pol oscillator is given by

$$\begin{aligned} \dot{x} &= y, \\ \dot{y} &= \mu \left(1 - x^2\right) y - x, \end{aligned} \tag{4.17}$$

where the nonlinear damping/gain parameter $\mu = 0.5$ is used for demonstration purposes. The system is simulated with initial condition $[-2, 1]$, $T = 10$, and $dt = 0.01$. The Adam optimizer with learning rate of 0.001 is used for all noise levels. The parameters of modified SINDy are chosen as $q = 1$ and $\lambda = 0.05$, and the library of candidate functions is constructed with polynomial terms up to third order (without constant term). Three

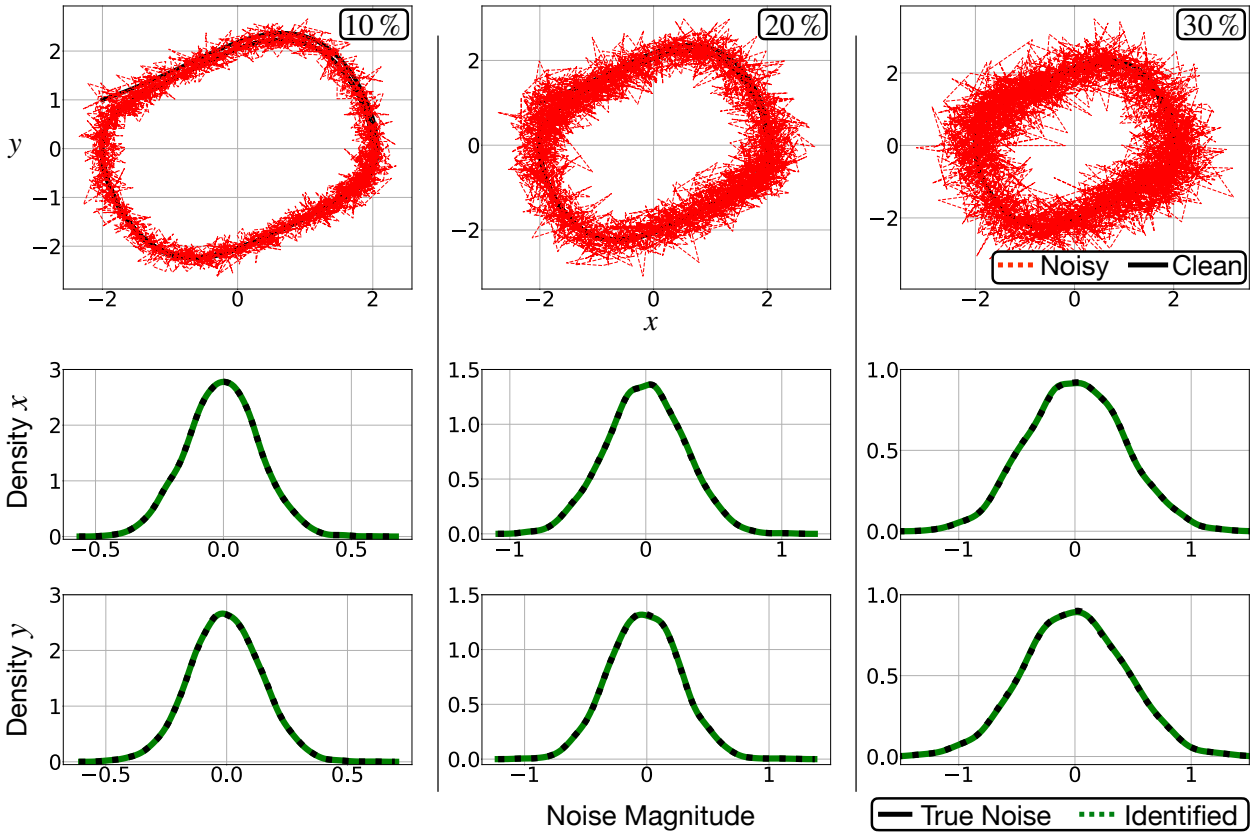


Figure 4.5: Distribution of the noise learned by modified SINDy is shown.

different levels of noise are applied and the distribution of identified noise is shown in Fig. 4.5. Fig. 4.5 shows that modified SINDy correctly identified the distribution of true noise.

4.4.2 Rössler Attractor

The second example we use is the Rössler attractor that is governed by

$$\begin{aligned} \dot{x} &= -y - z, \\ \dot{y} &= x + ay, \\ \dot{z} &= b + z(x - c), \end{aligned} \tag{4.18}$$

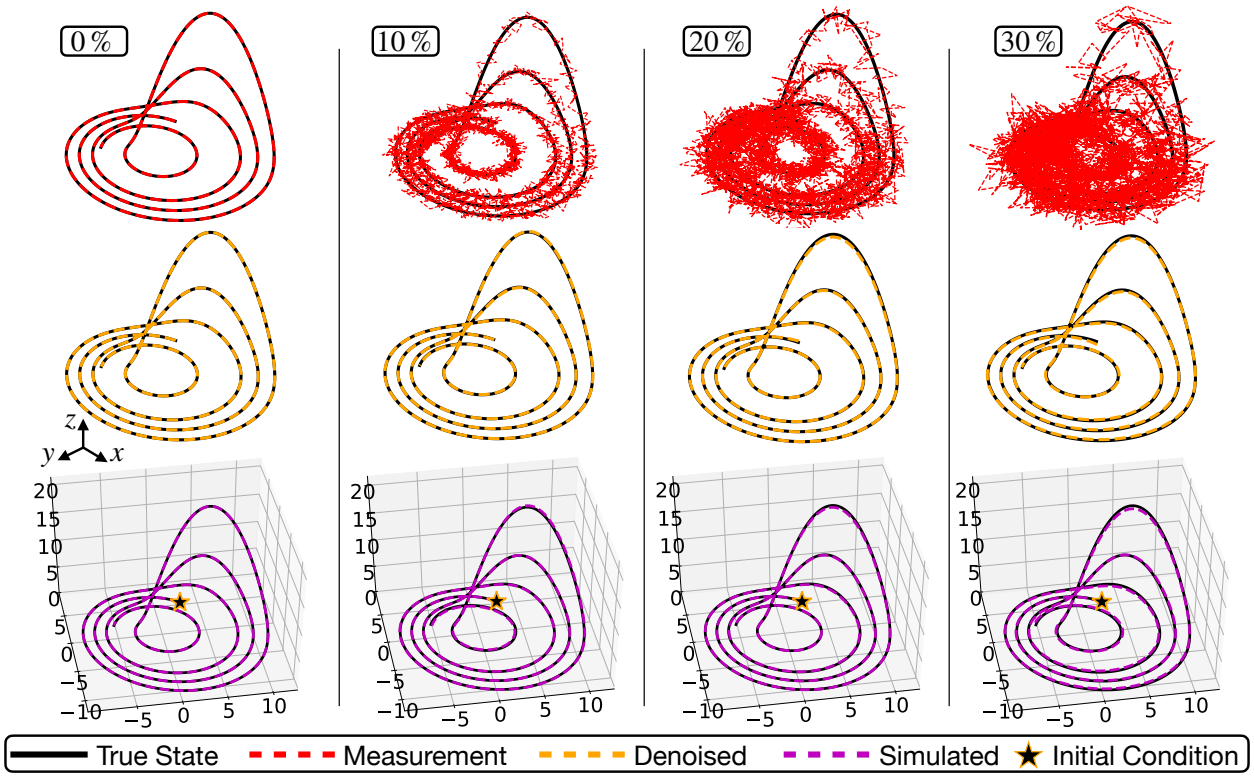


Figure 4.6: The modified SINDy is used to denoise the measurement of Rössler attractor while learning its model. The identified model shows high accuracy when simulating forward.

where $a = 0.2$, $b = 0.2$, and $c = 5.7$. The system is simulated with initial condition $[3, 5, 0]$, $T = 25$, and $dt = 0.01$. The Adam optimizer with learning rate of 0.001 is used for all noise levels. The parameters of modified SINDy are chosen as $q = 1$ and $\lambda = 0.05$, and the library of candidate functions is constructed with polynomial terms up to second order (with constant term). Three different levels of noise are applied and the denoised signal is shown in Fig. 4.6. Fig. 4.6 also shows the simulated trajectories of the identified models. The initial condition $[3, 5, 0]$, $T = 25$, and $dt = 0.01$ are used to simulate the identified models.

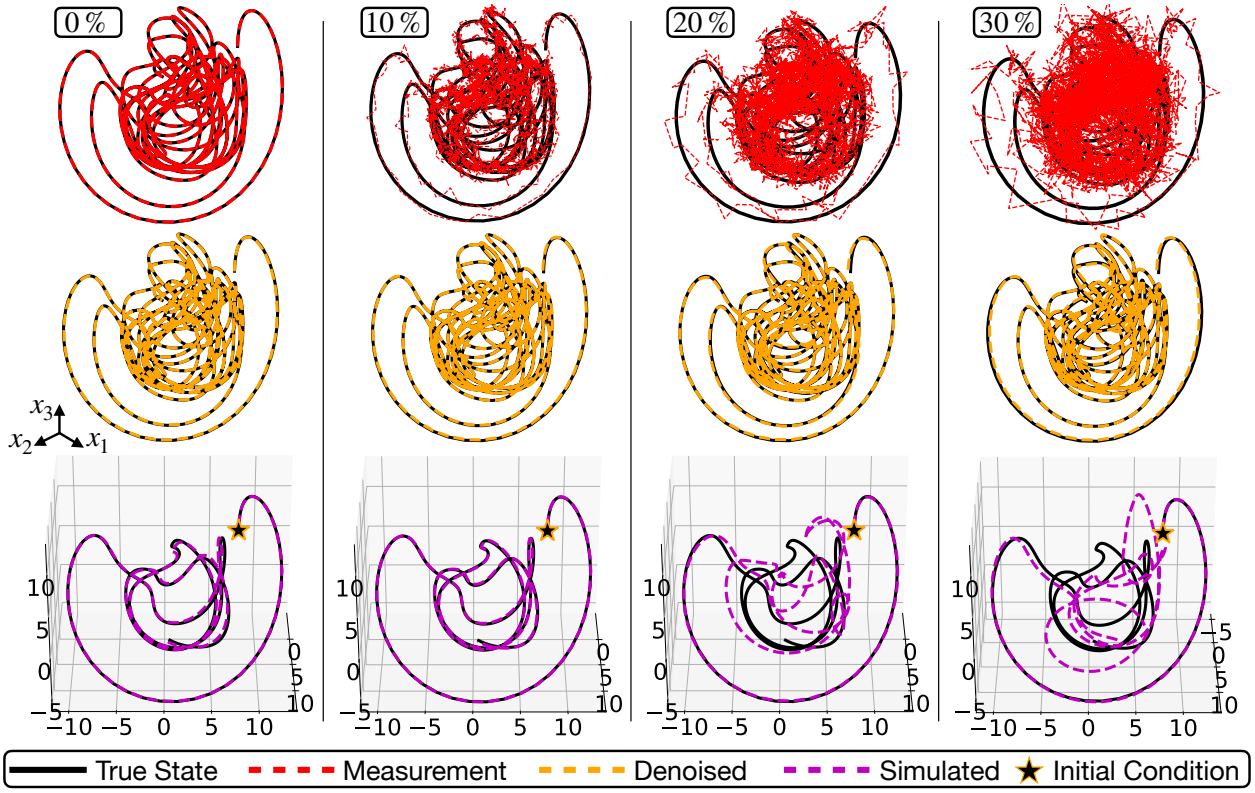


Figure 4.7: The modified SINDy is used to denoise the measurement of Lorenz 96 system while learning its model.

4.4.3 Lorenz 96 Model

As our last example, we use the modified SINDy to identify Lorenz 96 model whose equation is given by

$$\dot{x}_i = (x_{i+1} - x_{i-2})x_{i-1} - x_i + F, \tag{4.19}$$

for $i = 1, 2, \ldots, N$. We assume $x_{-1} = x_{N-1}$, $x_0 = x_N$, $x_1 = x_{N+1}$, and set forcing term F as 8 to generate chaotic behavior. The number N is set as 4 such that the model has 6 states. The system is simulated with initial condition $[1, 8, 8, 8, 8, 8]$, $T = 25$, and $dt = 0.01$. The Adam optimizer with learning rate of 0.001 is used for all noise levels. The parameters of modified SINDy are chosen as $q = 1$ and $\lambda = 0.1$ (for 30% noise, $\lambda = 0.05$). The library of candidate functions is constructed with polynomial terms up to third order (with constant

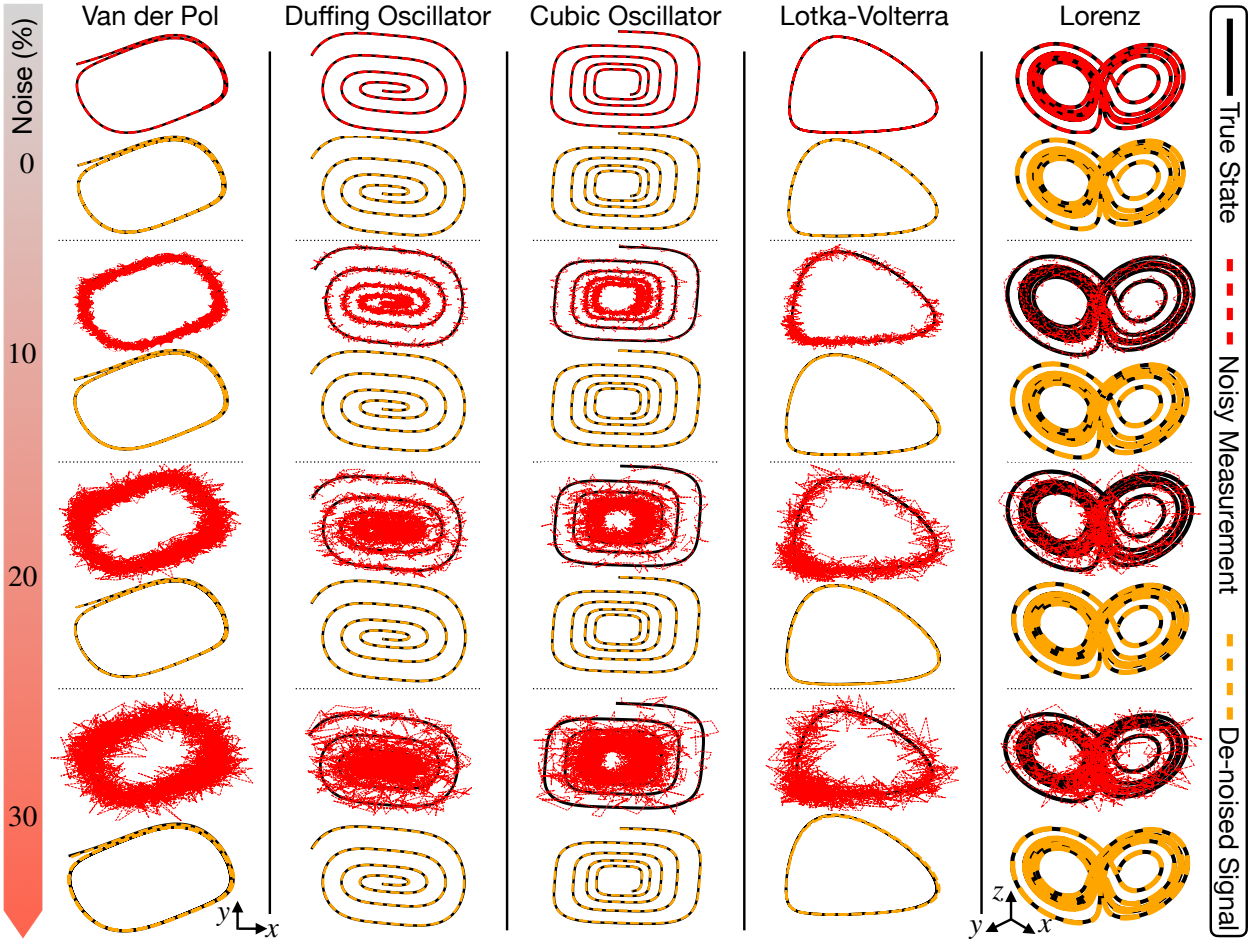


Figure 4.8: This plot shows the denoising ability of modified SINDy with different examples. Regardless of the noise percentage, modified SINDy correctly identified all the system models.

term included, 84 candidates in total). Three different levels of noise are applied and the denoised signal is shown in Fig. 4.7 (for ease of visualization, only the first three states are shown). Fig. 4.7 also shows the simulated trajectories of identified models. The initial condition $[1, 8, 8, 8, 8, 8]$, $T = 5$, and $dt = 0.01$ are used to simulate the identified models.

In Fig. 4.8, the effectiveness of modified SINDy is demonstrated on a number of canonical dynamical systems models. For all examples, Gaussian noise with zero-mean is added to generate the noisy training data, and Adam optimizer is used to perform the

optimization. The models and other parameters used for each example are summarized in Appendix. B.6. The modified SINDy correctly identified all the system model and noise distribution regardless of the noise magnitude used.

4.4.4 Identification of Noise Distributions

The modified SINDy algorithm has the ability to to handle different kinds of noise distributions. Three different kinds of noise distributions are used to demonstrate this: Gaussian, Uniform, and Gamma. To generate the Gamma noise, its shape and scale are set to 1. The generated noise is multiplied by Noise Percentage $\times$ var(Signal). The noise-free data of Van der Pol oscillator is generated the same way in Sec. 4.4.1. The prediction step is set to $q = 2$ and the sparsity parameter is set to $\lambda = 0.15$. Fig. 4.9 shows the distribution identified by modified SINDy. Fig. 4.9 shows that learning the non-zero mean noise distribution is more difficult than learning a zero-mean one. For better learning results of a non-zero mean noise distribution, one can try the iterative learning approach shown in Appendix. B.8. Once the noise is separated from the signal, an additional step can be taken to identify the distribution of noise from the candidate distributions. This can be achieved by the `fitter` package in Python [342]. Appendix. B.9 shows more details of this process.

4.5 Conclusion and Future Work

In this work, we introduce a new learning algorithm that leverages automatic differentiation and sparse regression for simultaneously (i) denoising time-series data, (ii) learning and parametrizing the noise probability distribution, and (iii) identifying the underlying parsimonious dynamical system responsible for generating the time-series data. The method provides a critically enabling modification to the SINDy algorithm for improving robustness to noise with less training data in comparison with the previously developed NN denoising approach by Rudy et al. [318]. Multiple numerical examples

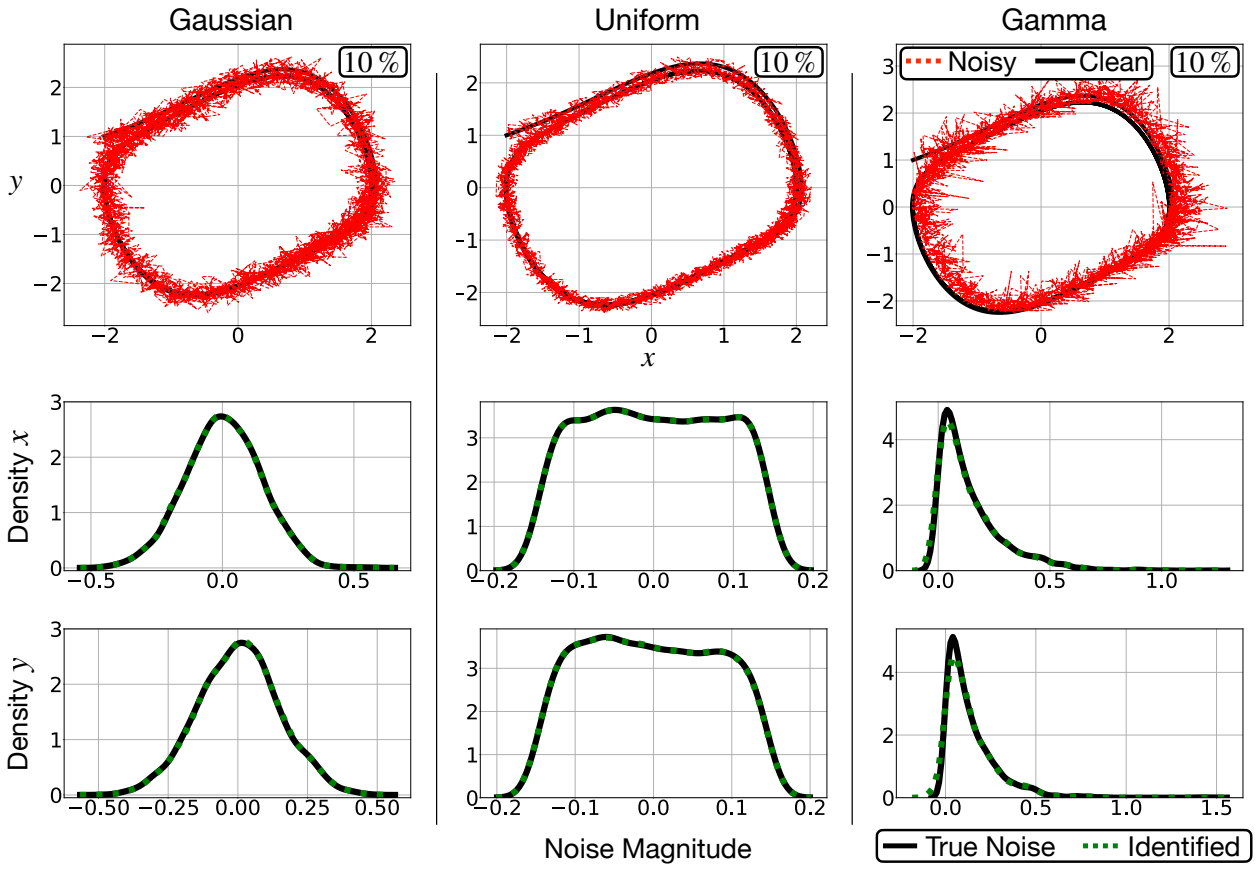


Figure 4.9: The ability of modified SINDy to handle different kinds of noise distribution is illustrated in this figure, and the Van der Pol oscillator is used as an example. 10% of noise is generated and added to the clean signal. As this figure shows, modified SINDy can identify different types of noise distribution correctly.

are shown to demonstrate the effectiveness of the modified SINDy method for signal and noise separation as well as model identification. Importantly, we have shown that modified SINDy can be used to learn various types of noise distributions, including Gaussian, uniform, and non-zero mean noise distributions, such as a Gamma distribution. Overall, the modified SINDy is a robust method with practical potential for handling highly noisy data sets and/or when partial model information is known.

The modified SINDy is modular, allowing for many easily integrated improvements. An important direction for development includes the incorporation of control inputs,

since many systems of practical interest are actuated, such as the pendulum on a cart system [177, 280]. Extending modified SINDy to consider the impact of control will significantly expand its application domain. Moreover, it is also desirable to incorporate the Weak formulation into the current framework. As shown in [232–234, 309], the parameter error can be improved when a compact smooth support function is included into Eq. (4.5). In current framework, the choice of support function in Eq. (4.5) is $\omega = 1$, which is suboptimal. Thus, including the Weak formulation into current framework can have great potential to improve the parameter error. Improvements in computational speed are also desirable. In comparison with the sequential least-square thresholding of the standard SINDy algorithm, the Adam optimizer is slow. There is the potential to use the standard SINDy sparse regression algorithms to warm start the Adam optimization routine. There also exist possibility to use alternative packages like JAX MD [338, 339] to perform automatic differentiation, and it would be interesting to see the speed improvement it can achieve compared to Tensorflow. The modified SINDy can also be integrated with SINDy-PI to identify rational or implicit dynamics, which is quite difficult since the simulation error shown in Eq. (4.8) can not be calculated easily when the dynamics take a rational form. In order to denoise the signal generated from rational system, the implicit ODE solver needs to be used in order to simulate the implicit dynamics forward and backward in time. Currently, we do not have an elegant way to implement the implicit ODE solver in Tensorflow, and our future research includes extension of modified SINDy to identify rational dynamics. This is the case where the use of the NN denoising approach [318] by Rudy et al. is ideal.

Finally, it is important to improve the robustness of the modified SINDy algorithm when a large number of library terms are used. Currently, the modified SINDy can not handle large libraries robustly due to the non-convexity of the optimization problem. When the library is too large, the problem becomes unstable without decreasing the optimizer's learning rate. One potential solution is to simulate the dynamics with a variable time step numerical simulation scheme instead of a fixed step scheme, as we used in this paper.

Although there are still many improvements to be made, we believe the introduction of modified SINDy will help guide the use of automatic differentiation tools to improve the SINDy framework.

Chapter 5

LEARNING DISCREPANCY MODELS FROM EXPERIMENTAL DATA

5.1 Introduction

In this Chapter, we propose a hybrid modeling approach, where we employ data-driven techniques to model the discrepancy between a simplified or insufficient physical model and observed measurements. We employ the sparse identification of nonlinear dynamics (SINDy) framework [180] to discover parsimonious and interpretable discrepancy models. We also show how modified SINDy can be used in the discrepancy modeling framework. Thus, we leverage prior knowledge of the simplified physics, while more accurately modeling details of the true system.

As an illustrative example for this paper, we consider a double pendulum on a cart experiment. We typically model this type of experiment as a simple mechanical system with a few degrees of freedom, described by either a Hamiltonian or Lagrangian [343]. However, this model neglects real-world effects such as nonlinear friction, bearing chatter, and wind resistance. These factors may all reduce control performance, for example, in model predictive control (MPC) [281, 344, 345], where prediction errors adversely affect robustness [346]. For this specific system, designing a feed-forward control law to swing up the pendulum from rest requires a highly accurate model. Even a small mismatch in the system model may result in a considerable deviation in the computed trajectory, since it is a chaotic system. It is also challenging to obtain a data-driven model of this system with the correct structure, so it is beneficial to incorporate prior physical knowledge in the form of a simplified Hamiltonian or Lagrangian.

5.1.1 *Problem statement: Discrepancy modeling*

There are several reasons why model discrepancies occur [347, 348]. First, there may be measurement noise and exogenous disturbances. In this case, the Kalman filter may be thought of as a discrepancy model where the mismatch between a simplified model and observations is assumed to be a Gaussian process [349]. Next, the parameters of the system may be inaccurately modeled. Even worse, the structure of the model may not be correct, either because important terms are missing or erroneous terms are present. This is known as *model inadequacy* or model structure mismatch. Other challenges include incomplete measurements and latent variables, delays, and sensitive dependence on initial data in chaotic systems.

In this Chapter, we focus on parameter and structural uncertainties, although a broader framework is the subject of ongoing work. We consider dynamical systems of the form

$$\frac{d}{dt}\mathbf{x} = \boldsymbol{f}(\mathbf{x}, \mathbf{u}; \boldsymbol{\mu}), \tag{5.1}$$

where $\mathbf{x} \in \mathbb{R}^{1\times n}$ is the state, $\mathbf{u} \in \mathbb{R}^{1\times r}$ is the control input, $\boldsymbol{\mu} \in \mathbb{R}^{p}$ are the parameters, and $\boldsymbol{f}$ are the dynamics. We assume full-state measurements, although generally the state must be estimated from limited measurements.

The discrepancy modeling problem seeks to model the difference between a quantity of interest $\boldsymbol{\phi}(t)$ from a physical model $\boldsymbol{\phi}_m(t)$ and the observed value $\boldsymbol{\phi}_o(t)$:

$$\delta\boldsymbol{\phi}(t) = \boldsymbol{\phi}_o(t) - \boldsymbol{\phi}_m(t), \tag{5.2}$$

where $\delta\boldsymbol{\phi}$ is the discrepancy. Here, we consider the quantity of interest $\boldsymbol{\phi}(t)$ to be the dynamics themselves, or more precisely the rate of change of the state in time. However, this framework is more general and can incorporate several other forms of model discrepancy. For example, $\boldsymbol{\phi}(t)$ may be a conserved quantity, such as the Hamiltonian, from which the dynamics are derived.

5.1.2 *Related work*

Accurate system modeling is a critical task in science and engineering, including for autonomous robotics [350, 351] and process control [352–354]. A model that is able to accurately predict the future state of a system is imperative for prediction and control. However, the high-dimensional, nonlinear, and multi-scale nature of many systems renders modeling a challenging task.

System identification has reached a high degree of maturity encompassing myriad techniques to identify linear and nonlinear systems [56, 355] from data, including state-space modeling via the eigensystem realization algorithm (ERA) [158] and other subspace identification methods, Volterra series [356, 357], linear and nonlinear autoregressive models [69] (e.g., ARX, ARMA, NARX, and NARMAX), and neural network models [358, 359], to name only a few. Machine learning techniques such as manifold learning and non-parametric modeling have also been useful for identifying nonlinear systems [360, 361].

There is an increasing shift from black-box modeling to developing models that are physically intuitive and interpretable, as well as models that are constrained models with known prior information. For instance, genetic programming can be used to infer governing equations from data [13, 14]. The recent SINDy approach [180] identifies parsimonious and interpretable models by promoting sparsity, and it has been extended to incorporate the effect of control [281] and to take into account expert knowledge, such as symmetries and conservation laws [254]. However, it may be challenging to identify certain types of systems, such as those containing rational function nonlinearities [181]. Instead of modeling the system entirely from data, when partial information is available, such as an idealized Hamiltonian, data-driven modeling procedures may focus only on modeling the discrepancy.

One way to compensate for model discrepancy is with Bayesian approaches [347], where a model discrepancy function is learned and the model output uncertainty is quantified from data [362]. However, this method requires the selection of a prior form for the

model discrepancy function, which is difficult due to the lack of physical knowledge about the model structure [363]. Furthermore, this method may potentially introduce bias in the model parameters if calibration is performed without knowledge of the model discrepancy [364, 365]. Alternatively, reinforcement learning can be used to learn the mismatch between a model and the actual dynamics, for example in robotics [9]. The learned dynamics are then used with the conceptual model to control the real system. However, neither Bayesian methods nor reinforcement learning can provide an interpretable representation for the model mismatch, and thus conceal the physical meaning of the discrepancy model discovered.

5.1.3 Contributions of this work

In the present work, we leverage SINDy to compensate for model parameter and structure mismatch given an imperfect model of the system. This serves several purposes: (1) Prior partial knowledge on the system or a previously learned model may be available and can be incorporated to aid the modeling process and improve prediction accuracy. (2) The SINDy algorithm suffers from the curse of dimensionality due to the growth of the library size with increasing number of variables in the system, which makes it challenging to discover the full governing equations when a large library size is required. However, learning the model mismatch significantly reduces this burden, focusing SINDy only on modeling the mismatch. (3) Learning interpretable representations of the model mismatch may inform physical intuition and generalize beyond the training data. We will demonstrate that SINDy is able to model discrepancies such as incorrect system parameters and model inadequacy (or structure) errors. The learned SINDy discrepancy model can then be used to enhance the imperfect model, providing an improved description of the system dynamics.

5.2 Data-Driven Discovery of Model Discrepancy

This section applies the discrepancy modeling framework using SINDy to the dynamical system shown in Eq. (5.1). We skip the details of the SINDy algorithm as it is already discussed in Sec. 2.1.

5.2.1 Discrepancy modeling for systems without control

Although it is possible to model any physical quantity, we consider modeling the dynamics themselves [180]. Consider noisy measurements from a true dynamical system $\boldsymbol{f}$

$$\boldsymbol{\phi}_o(t) = \boldsymbol{f}(\mathbf{x}(t); \boldsymbol{\mu}) + \epsilon, \tag{5.3}$$

where $\epsilon \in \mathbb{R}^n$ is the measurement noise. The model output for this system is represented as

$$\boldsymbol{\phi}_m(t) = \boldsymbol{f}_m(\mathbf{x}(t); \boldsymbol{\mu}_1). \tag{5.4}$$

Parameter error, $\boldsymbol{\mu} \neq \boldsymbol{\mu}_1$, model inadequacy, $\boldsymbol{f} \neq \boldsymbol{f}_m$, and measurement error will cause a model mismatch, given by

$$\delta\boldsymbol{\phi}(t) = \boldsymbol{\phi}_o(t) - \boldsymbol{\phi}_m(t) = \boldsymbol{f}(\mathbf{x}(t); \boldsymbol{\mu}) - \boldsymbol{f}_m(\mathbf{x}(t); \boldsymbol{\mu}_1). \tag{5.5}$$

We then construct a model for the discrepancy $\delta\boldsymbol{\phi}(t)$ as a function of the state and new parameters $\boldsymbol{\mu}_2$, capturing the parameter and structure mismatch:

$$\delta\boldsymbol{\phi}(t) = \boldsymbol{g}(\mathbf{x}(t); \boldsymbol{\mu}_2). \tag{5.6}$$

Model error data is collected in

$$\delta\boldsymbol{\Phi} = [\delta\boldsymbol{\phi}(t_1); \delta\boldsymbol{\phi}(t_2); \cdots; \delta\boldsymbol{\phi}(t_m)], \tag{5.7}$$

where $\delta\boldsymbol{\Phi} \in \mathbb{R}^{m\times n}$.

Given data $\delta\boldsymbol{\Phi}$ and $\mathbf{X}$, we use SINDy to sparsely represent $\boldsymbol{g}(\mathbf{x}(t);\boldsymbol{\mu}_2)$ in a library of candidate functions $\boldsymbol{\Theta}(\mathbf{X})$. Constructing this library typically requires some prior knowledge of the system to select a suitable basis in which the discrepancy model will be sparse. The sparse regression problem is then formulated as

$$\delta\boldsymbol{\Phi} = \boldsymbol{\Theta}(\mathbf{X})\boldsymbol{\Xi}. \tag{5.8}$$

Realistically, we will only have access to measurements of $\mathbf{x}(t)$, from which we may derive $\frac{d}{dt}\mathbf{x}(t)$, and the quantity of interest becomes $\boldsymbol{\phi}(t) \approx \frac{d}{dt}\mathbf{x}(t)$. The model output is $\boldsymbol{\phi}_m = \boldsymbol{f}_m(\mathbf{x};\boldsymbol{\mu}_1)$, the discrepancy is $\delta\boldsymbol{\phi} = \frac{d}{dt}\mathbf{x}(t) - \boldsymbol{f}_m(\mathbf{x};\boldsymbol{\mu}_1)$, and we have

$$\delta\dot{\mathbf{X}} := \dot{\mathbf{X}} - \boldsymbol{f}_m(\mathbf{X};\boldsymbol{\mu}_1) = \boldsymbol{\Theta}(\mathbf{X})\boldsymbol{\Xi}. \tag{5.9}$$

We solve for the sparsest coefficient matrix $\boldsymbol{\Xi}$ that satisfies Eq. (5.9) using the SINDy approach. A schematic of the method is displayed in Fig. 5.1.

5.2.2 Discrepancy modeling for systems with control

We now extend the formulation above to identify dynamics that are affected by a control input $\mathbf{u}(t) \in \mathbb{R}^q$:

$$\boldsymbol{\phi}_o(t) = \boldsymbol{f}(\mathbf{x}(t),\mathbf{u}(t);\boldsymbol{\mu}) + \epsilon, \tag{5.10}$$

The model output for this system is represented as

$$\boldsymbol{\phi}_m(t) = \boldsymbol{f}_m(\mathbf{x}(t),\mathbf{u}(t);\boldsymbol{\mu}_1). \tag{5.11}$$

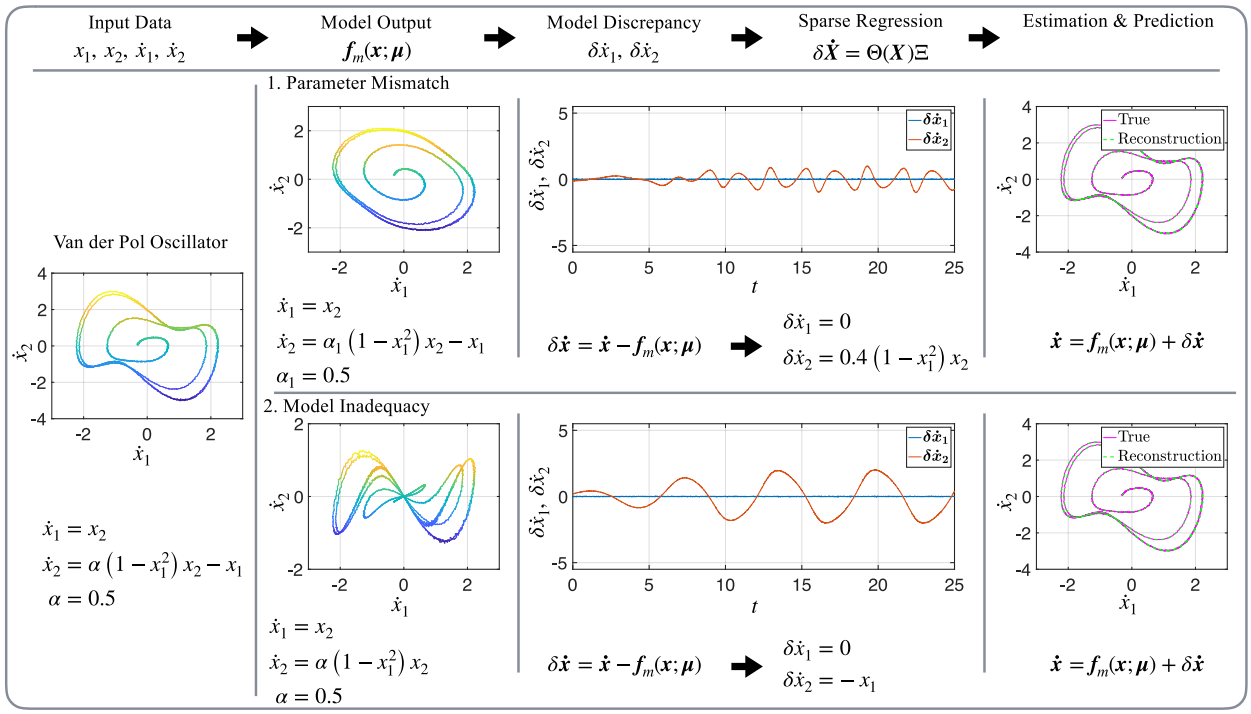


Figure 5.1: Illustration of SINDy to discover the system-model mismatch for the Van der Pol oscillator. The system is simulated to generate the measurement data. Measurement data x_1 and x_2 are provided to calculate the output estimated by our imperfect model. Sparse regression is used to infer the discrepancy model for the difference between the actual output and estimated output. The discrepancy model is then combined with the imperfect model to provide a better estimation of system dynamics. The model is then cross-validated on a new initial condition $\mathbf{x}_0 = (-0.2, -0.3)$. As we can see, the model discovered by SINDy is able to compensate for the discrepancy between the actual system and flawed model.

As with the uncontrolled case, due to the model inadequacy or parameter error, there will be a discrepancy

$$\begin{aligned} \delta\boldsymbol{\phi}(t) &= \boldsymbol{\phi}_o(t) - \boldsymbol{\phi}_m(t) \\ &= \boldsymbol{f}(\mathbf{x}(t), \mathbf{u}(t); \boldsymbol{\mu}) - \boldsymbol{f}_m(\mathbf{x}(t), \mathbf{u}(t); \boldsymbol{\mu}_1). \end{aligned} \tag{5.12}$$

Our goal is to model $\delta\boldsymbol{\phi}(t)$ and we further assume it is a function of the system state $\mathbf{x}$, control input $\mathbf{u}$, and new parameters $\boldsymbol{\mu}_2$:

$$\delta\boldsymbol{\phi}(t) = \boldsymbol{g}(\mathbf{x}(t), \mathbf{u}(t); \boldsymbol{\mu}_2). \tag{5.13}$$

To achieve this goal, we form a sparse regression problem

$$\delta\boldsymbol{\Phi} = \boldsymbol{\Theta}(\mathbf{X}, \mathbf{U})\boldsymbol{\xi}, \tag{5.14}$$

where $\mathbf{U} \in \mathbb{R}^{m\times r}$ is

$$\mathbf{U} = [\mathbf{u}(t_1); \mathbf{u}(t_2); \cdots; \mathbf{u}(t_m)]. \tag{5.15}$$

As before, we will actually be observing $\boldsymbol{\phi}(t) \approx \frac{d}{dt}\mathbf{x}(t)$ with model $\boldsymbol{\phi}_m = \boldsymbol{f}_m(\mathbf{x}, \mathbf{u}; \boldsymbol{\mu}_1)$. The discrepancy becomes $\delta\boldsymbol{\phi} = \frac{d}{dt}\mathbf{x}(t) - \boldsymbol{f}_m(\mathbf{x}, \mathbf{u}; \boldsymbol{\mu}_1)$, resulting in the following regression problem:

$$\delta\dot{\mathbf{X}} := \dot{\mathbf{X}} - \boldsymbol{f}_m(\mathbf{X}, \mathbf{U}; \boldsymbol{\mu}_1) = \boldsymbol{\Theta}(\mathbf{X}, \mathbf{U})\boldsymbol{\Xi}. \tag{5.16}$$

Note that the library $\boldsymbol{\Theta}(\mathbf{X}, \mathbf{U})$ has the form

$$\boldsymbol{\Theta}(\mathbf{X}, \mathbf{U}) = [\theta_1(\mathbf{X}, \mathbf{U})\ \theta_2(\mathbf{X}, \mathbf{U})\ \cdots \theta_v(\mathbf{X}, \mathbf{U})], \tag{5.17}$$

where $\theta_i(\mathbf{X}, \mathbf{U}) \in \mathbb{R}^{m\times 1}$ is a candidate function that may explain the discrepancy $\delta\boldsymbol{\phi}(t)$. The functions $\theta_i(\mathbf{X}, \mathbf{U})$ can be any combination of $\mathbf{X}$ and $\mathbf{U}$. For example, $\theta_i(\mathbf{X}, \mathbf{U}) = \sin(\mathbf{X})\cos(\mathbf{U})$, $\theta_i(\mathbf{X}, \mathbf{U}) = \mathbf{X}\mathbf{U}^2$. By solving Eq. (5.14) we are able to model $\boldsymbol{g}(\mathbf{x}(t), \mathbf{u}(t); \delta\boldsymbol{\mu}_2)$.

5.3 Examples

In this section, we demonstrate applications of the proposed approach to discover model discrepancies. We start with an illustrative example and then apply this method to experimental data from a double pendulum on a cart. We also show how modified SINDy can be used in the discrepancy modeling framework to account for highly noisy data.

5.3.1 *Van der Pol oscillator*

To begin, we will focus on an illustrative example, the Van der Pol oscillator, and show how the SINDy method can be used to compensate for both parameter errors and model inadequacy. The Van der Pol oscillator is given by

$$\dot{x}_1 = x_2, \tag{5.18a}$$

$$\dot{x}_2 = \alpha \left(1 - x_1^2\right) x_2 - x_1, \tag{5.18b}$$

with parameter $\alpha = 0.5$.

Parameter mismatch

Suppose we do not know the true parameter α, but instead have an approximation $\alpha_1 = 0.1$. It is possible to compensate for this model discrepancy caused by parameter mismatch. We first gather the measurement data of the actual system, in this case by integrating the true dynamics using a fourth order Runge Kutta scheme. We integrate for 25 time units with time step $\Delta t = 0.01$ and initial condition is $\mathbf{x}_0 = (0.5, 0)$. Random Gaussian measurement noise with amplitude 0.01 is added to the data.

We evaluate the model dynamics $\boldsymbol{f}(\mathbf{x}(t); \alpha_1)$, with the measured trajectory $\mathbf{x}(t)$ and the inaccurate parameter α_1. The discrepancy between the true system output and the model is then given by $\delta\dot{\mathbf{x}}(t) := \frac{d}{dt}\mathbf{x}(t) - \boldsymbol{f}(\mathbf{x}(t); \alpha_1)$. The augmented error matrix is formed as

$$\delta\dot{\mathbf{X}} = \left[\delta\dot{\mathbf{x}}(t_0); \quad \delta\dot{\mathbf{x}}(t_1); \quad \delta\dot{\mathbf{x}}(t_2); \quad \dots \quad \delta\dot{\mathbf{x}}(t_m)\right], \tag{5.19}$$

and the augmented state is

$$\mathbf{X} = \left[\mathbf{x}(t_0); \quad \mathbf{x}(t_1); \quad \mathbf{x}(t_2); \quad \dots \quad \mathbf{x}(t_m)\right]. \tag{5.20}$$

The next step is to evaluate the library on the data. Here, we construct the library as

$$\boldsymbol{\Theta}(x_1, x_2) = \begin{array}{c} \text{Time} \\ \downarrow \end{array} \begin{array}{c} \xrightarrow{\qquad\qquad\text{Functions}\qquad\qquad} \\ \begin{bmatrix} | & | & | & | & | & | & | \\ 1 & x_1 & x_2 & x_1x_2 & x_1^2x_2 & x_1x_2^2 & x_1^2x_2^2 \\ | & | & | & | & | & | & | \end{bmatrix} \end{array}, \tag{5.21}$$

with polynomial terms up to second order. Finally, we form the sparse regression problem

$$\delta\dot{\mathbf{X}} = \boldsymbol{\Theta}(x_1, x_2)\boldsymbol{\Xi}, \tag{5.22}$$

and solve for the coefficient matrix $\boldsymbol{\Xi}$. Results are shown in the top row of Fig. 5.1. SINDy successfully identifies the model discrepancy.

Model Inadequacy (or structure mismatch)

We now assume that the model discrepancy is caused by model inadequacy, where the model $\boldsymbol{f}_m$ is missing the linear term in the second equation of the Van der Pol system:

$$\dot{x}_1 = x_2, \tag{5.23a}$$

$$\dot{x}_2 = \alpha\left(1 - x_1^2\right)x_2. \tag{5.23b}$$

We assume that the parameter $\alpha = 0.5$ is correct. The data, $\mathbf{x}, \dot{\mathbf{x}}, \delta\dot{\mathbf{x}}$ is collected and the library of functions is constructed as before. The sparse coefficient matrix $\boldsymbol{\Xi}$ is determined via SINDy, and the model discrepancy is successfully modeled as shown in the bottom row of Fig. 5.1.

To summarize, the SINDy method successfully identifies the discrepancy between the underlying governing equation and an inaccurate model. The imperfect model is then combined with the SINDy discrepancy model, resulting in an accurate prediction of the true system dynamics.

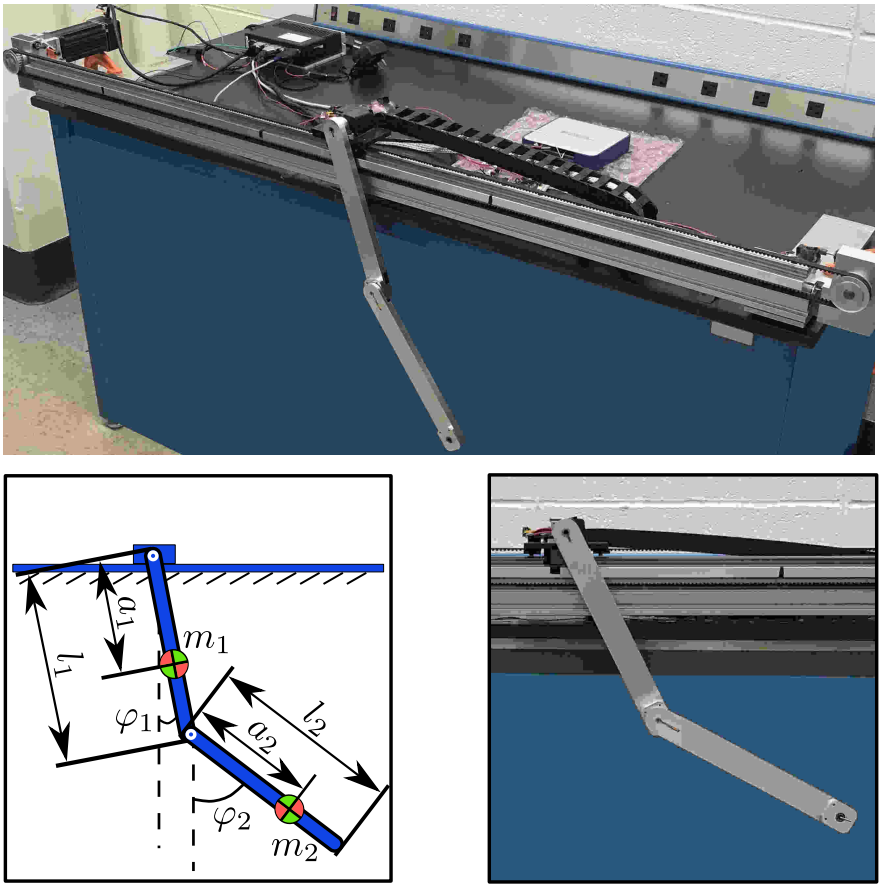


Figure 5.2: Double pendulum on a cart system with schematic (bottom left) and zoom of the two-link pendulum (bottom right). The pendulum cart is locked at its position to prevent horizontal movement.

5.3.2 Experimental double pendulum on a cart

We now demonstrate the use of SINDy to identify the time-dependent Hamiltonian function for the double pendulum on a cart from experimental data. A non-dissipative pendulum is a quintessential example of a conservative system, which admits invariants such as the Hamiltonian function, from which the governing equations can be derived. The conservative Hamiltonian is given by $H_c(\boldsymbol{q}, \boldsymbol{p}) = T + V$, where $\boldsymbol{q}$ and $\boldsymbol{p}$ are the generalized position and momentum of the system, and T and V represent the kinetic and potential energy of the system; the Hamiltonian H_c is the total energy, which is constant along a

trajectory. Since we measure $\boldsymbol{q}$ and compute the time derivative $\dot{\boldsymbol{q}}$, we will represent H_c as a function of $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$ in the following.

Real-world mechanical systems are, however, generally not conservative, but instead exhibit friction and damping from joints and wind resistance. Thus, the total energy is not conserved, and instead decays over time without additional exogenous energy input. While the potential and kinetic energy terms can be easily formulated, dissipation terms can be more challenging to derive. In this Chapter, we seek to identify the time-dependent dissipative effects, by modeling the difference between the conservative model Hamiltonian and the measured energy of the system. In practice, the *observed* energy is obtained by evaluating the idealized conservative Hamiltonian, consisting of the kinetic and potential terms T and V, on the measured trajectory. The *model* energy is given by evaluating the idealized Hamiltonian on the initial condition. Thus, the difference gives the energy dissipation, which we will model with SINDy:

$$\delta H(\boldsymbol{q}(t), \dot{\boldsymbol{q}}(t)) := H_m(\boldsymbol{q}(t), \dot{\boldsymbol{q}}(t)) - H_m(\boldsymbol{q}(0), \dot{\boldsymbol{q}}(0)). \tag{5.24}$$

Problem Formulation

We consider the double pendulum on a cart as shown in Fig. 5.2. The kinetic and potential energy of the double pendulum (assuming a locked cart position) are

$$\begin{aligned} T &= \frac{1}{2}(m_1(\dot{x}_1^2 + \dot{y}_1^2) + m_2(\dot{x}_2^2 + \dot{y}_2^2)) \\ &\quad + \frac{1}{2}(I_1\dot{\varphi}_1^2 + I_2\dot{\varphi}_2^2), \end{aligned} \tag{5.25a}$$

$$V = (m_1 y_1 + m_2 y_2)g, \tag{5.25b}$$

where I_1 and I_2 are the inertia, m_1 and m_2 are the masses, and l_1 and l_2 are the lengths of each pendulum arm, respectively. φ_1 and φ_2 are the angles and $\dot{\varphi}_1$ and $\dot{\varphi}_2$ are the angular velocity of the first and second pendulum arm. The relative lengths to the center of mass of the first and second pendulum arm are given by a_1 and a_2.

The position of each pendulum arm's center of mass (x_1, y_1) and (x_2, y_2) in (5.25) can be determined as

$$x_1 = a_1 \sin(\varphi_1), \tag{5.26a}$$

$$x_2 = l_1 \sin(\varphi_1) + a_2 \sin(\varphi_2), \tag{5.26b}$$

$$y_1 = a_1 \cos(\varphi_1), \tag{5.26c}$$

$$y_2 = l_1 \cos(\varphi_1) + a_2 \cos(\varphi_2). \tag{5.26d}$$

We assume that we can accurately determine the kinetic and potential energy of the system

$$H_m(\varphi_1, \varphi_2, \dot{\varphi}_1, \dot{\varphi}_2) = T + V, \tag{5.27}$$

which represents the insufficient model for the total energy. The discrepancy model $\delta H(\varphi_1, \varphi_2, \dot{\varphi}_1, \dot{\varphi}_2)$ then comprises the dissipative energy terms. The frictional torque of the pendulum arm can be modeled as $\Gamma_1 = k_1 \dot{\varphi}_1$ and $\Gamma_2 = k_2(\dot{\varphi}_1 - \dot{\varphi}_2)$, where k_1 and k_2 are damping coefficients. Then, $\delta H(\varphi_1, \varphi_2, \dot{\varphi}_1, \dot{\varphi}_2)$ is given by

$$\begin{aligned} \delta H(\varphi_1, \varphi_2, \dot{\varphi}_1, \dot{\varphi}_2) &= \int_0^t \Gamma_1 \dot{\varphi}_1 + \Gamma_2(\dot{\varphi}_1 - \dot{\varphi}_2) dt \\ &= \int_0^t k_1 \dot{\varphi}_1^2 + k_2(\dot{\varphi}_1 - \dot{\varphi}_2)^2 dt. \end{aligned} \tag{5.28}$$

In many engineering applications, the direct measurement of the frictional term is difficult if not impossible. Thus, we would like to use our data-driven approach to learn a model for the dissipated energy (5.28). Suppose that all the states $\varphi_1, \varphi_2, \dot{\varphi}_1, \dot{\varphi}_2$ can be measured or estimated, then $H_m(\varphi_1, \varphi_2, \dot{\varphi}_1, \dot{\varphi}_2)$ can be immediately calculated. Also, we can use the energy at the initial measurement time as reference for the total energy of the system, $H(\varphi_1(t_0), \varphi_2(t_0), \dot{\varphi}_1(t_0), \dot{\varphi}_2(t_0)) = E_0$. This allows us to calculate the dissipated energy as

$$\delta H(\varphi_1, \varphi_2, \dot{\varphi}_1, \dot{\varphi}_2) = E_0 - H_m(\varphi_1, \varphi_2, \dot{\varphi}_1, \dot{\varphi}_2). \tag{5.29}$$

Table 5.1: The parameters of the experimental double pendulum system.

Pendulum	Mass (kg)	Center of Mass (m)	Inertia ($kg \cdot m^2$)	Length (m)	Damping Coefficient	Gravity Constant (m/s^2)
1st Arm	0.2704	0.1910	0.003	0.2667	7.24×10^{-4}	9.818
2nd Arm	0.2056	0.1621	0.0011	0.2667	1.65×10^{-4}	

To model this energy discrepancy, we define a library $\boldsymbol{\Theta}(\varphi_1, \varphi_2, \dot{\varphi}_1, \dot{\varphi}_2)$ that contains polynomial and Fourier terms up to the third order. Then, the discrepancy can be represented by

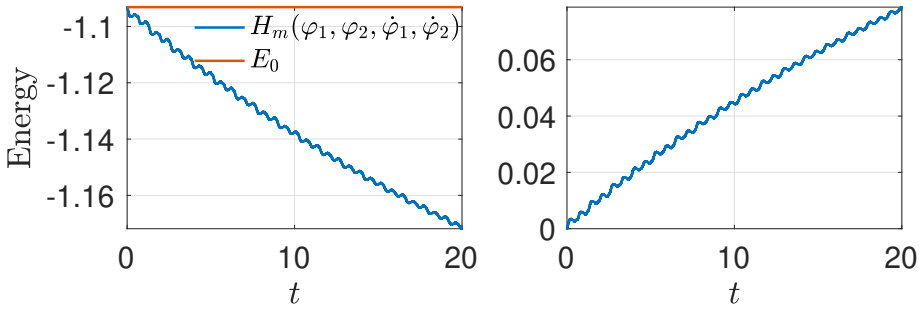
$$\delta H(\varphi_1, \varphi_2, \dot{\varphi}_1, \dot{\varphi}_2) = \boldsymbol{\Theta}(\varphi_1, \varphi_2, \dot{\varphi}_1, \dot{\varphi}_2)\boldsymbol{\xi}, \tag{5.30}$$

where $\boldsymbol{\xi}$ is a sparse vector that contains the coefficients of the active terms in the library.

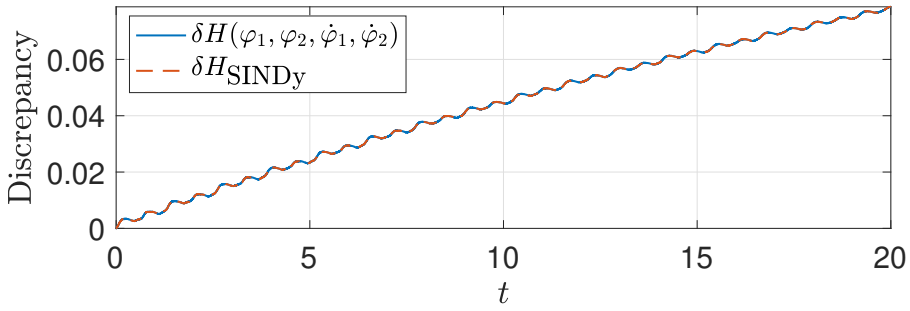
Results

Here, we present results for modeling the dissipation $\delta H(\varphi_1, \varphi_2, \dot{\varphi}_1, \dot{\varphi}_2)$. The parameters of our experimental system are displayed in Table. 5.1. All parameters are obtained using a parameter estimation technique [7]. The pendulum mass and length are constrained during the parameter estimation so that there won't be a large deviation from the measured value. The experiment was initialized at a random position, and the angles φ_1 and $\varphi_2 - \varphi_1$ were collected with a sampling rate of $\Delta t = 0.001$ for a duration of 20s. We used a US Digital HUBDISK-1 1" transmissive rotary encoder disk, which provides 5000 counts per revolution. Then the angular velocities $\dot{\varphi}_1$ and $\dot{\varphi}_2$ are approximated by taking numerical derivatives on the raw data. To mitigate noise, we first smooth the raw φ_1 and φ_2 data using the Savitzky-Golay filter and afterwards compute the derivative by numerical differentiation.

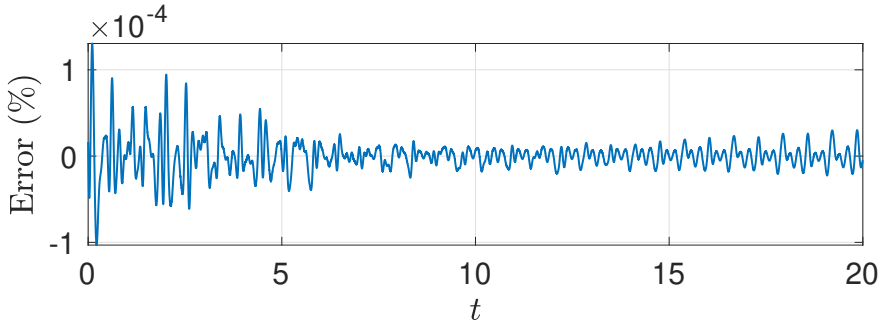
Results of the discrepancy model, identified with SINDy on the time series of $\delta H(\varphi_1, \varphi_2, \dot{\varphi}_1, \dot{\varphi}_2)$, are shown in Fig. 5.3 for the training data and in Fig. 5.4 for validation data. Note that in Fig. 5.3a the signal H_m does not decrease asymptoticly. We observe that the table on



(a) Left: Initial energy of the system E_0 and the sum of kinetic and potential energy $H_m(\varphi_1, \varphi_2, \dot{\varphi}_1, \dot{\varphi}_2)$, which decreases due to friction. Right: The difference between E_0 and $H_m(\varphi_1, \varphi_2, \dot{\varphi}_1, \dot{\varphi}_2)$ representing the energy dissipated by friction.



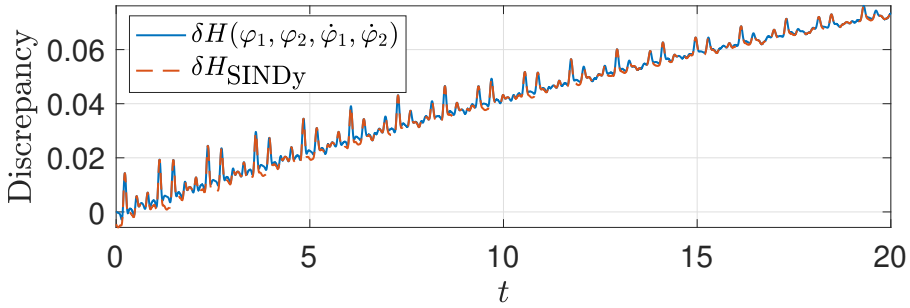
(b) The magnitude of energy dissipated by the friction $\delta H(\varphi_1, \varphi_2, \dot{\varphi}_1, \dot{\varphi}_2)$ and the SINDy identified dynamics. Note that the discrepancy is positive, since it is defined as the ideal conserved energy minus the measured dissipative energy.



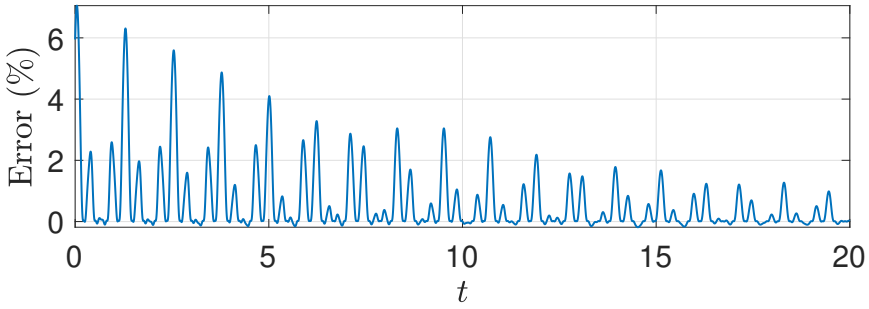
(c) The SINDy estimation error on the training data. The percentage is calculated using $\frac{\delta H(\varphi_1,\varphi_2,\dot{\varphi}_1,\dot{\varphi}_2) - \delta H_{\text{SINDy}}}{\max(|\delta H(\varphi_1,\varphi_2,\dot{\varphi}_1,\dot{\varphi}_2)|)} \times 100\%$.

Figure 5.3: Results for the identified model discrepancy caused by friction, demonstrated on training data.

which the pendulum in mounted oscillates with the pendulum, storing a small amount of potential energy. The error of the SINDy prediction and the measured $\delta H(\varphi_1, \varphi_2, \dot{\varphi}_1, \dot{\varphi}_2)$ is shown in Fig. 5.3b. From Fig. 5.3c we see that SINDy accurately describes the effect of



(a) The magnitude of energy dissipated by the friction $\delta H(\varphi_1, \varphi_2, \dot{\varphi}_1, \dot{\varphi}_2)$ and the SINDy estimation evaluated on validation data.



(b) The SINDy estimation error on validation data. The percentage is calculated using $\frac{\delta H(\varphi_1,\varphi_2,\dot{\varphi}_1,\dot{\varphi}_2)-\delta H_{\text{SINDy}}}{\max(|\delta H(\varphi_1,\varphi_2,\dot{\varphi}_1,\dot{\varphi}_2)|)} \times 100\%$.

Figure 5.4: Results for the identified model discrepancy caused by friction, demonstrated on validation data.

friction, explaining the model discrepancy.

Crossvalidation is critical to avoid overfitting. Hence, we test the identified model on a new validation dataset unseen during the training stage. The performance of the SINDy-discovered model on the validation data is shown in Fig. 5.4a. Although the errors are larger on the validation dataset, the SINDy model is quite accurate in modeling the missing friction term, demonstrating the ability of the discrepancy model to generalize to new test cases.

5.3.3 Double pendulum on a cart with control

In the previous section, SINDy is utilized to discover discrepancy models in the total energy caused by dissipation when given an insufficient conservative energy model. We

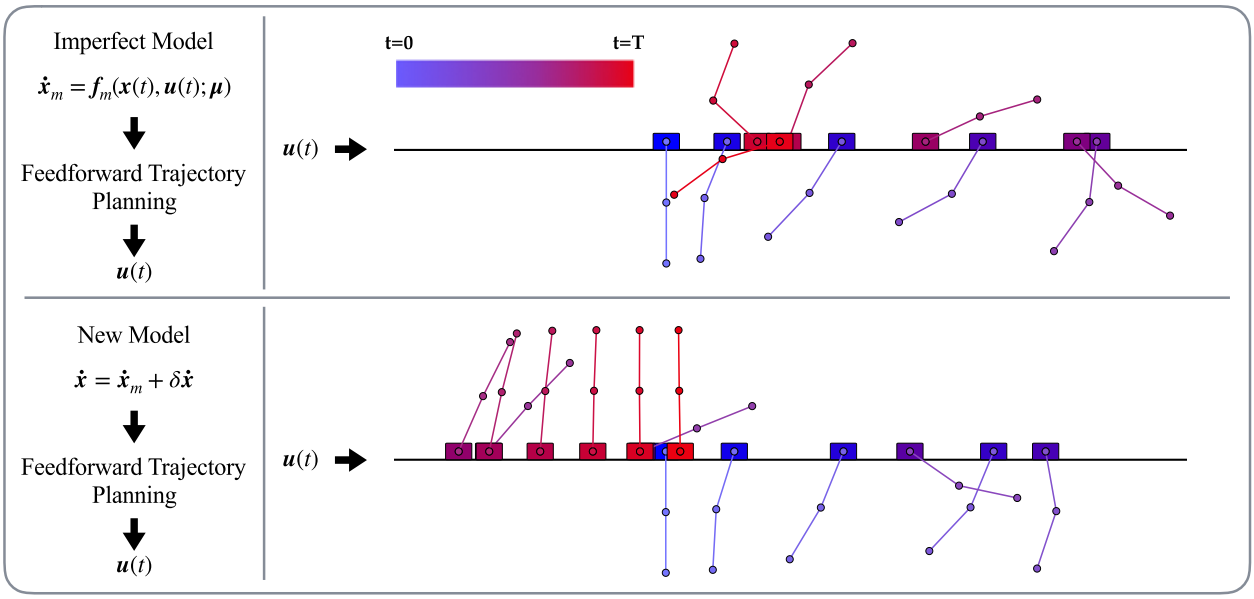


Figure 5.5: Swing-up of the double pendulum on a cart using feed-forward control. Top: The feed-forward control input is calculated using the imperfect model, which is in then applied to the actual system. The pendulum cannot reach the up-right position. Bottom: System's response under the control input calculated using the hybrid model, which is comprised of the imperfect model and the identified discrepancy model. Since the discrepancy is identified, the new model mimics the true dynamics of the double pendulum on a cart system.

now consider the case of an actuated double pendulum on a cart, where a control input is added to the system. This is an important generalization, since many real-life system are affected by control, which requires additional modeling. To discover the model discrepancy in systems with control, we follow the steps outlined in Sec. 5.2.2.

Problem formulation

In this example, we consider data from a numerical simulation of the double pendulum on a cart, shown in Fig. 5.2, with parameters provided in Tab. 5.1. The equations of motion can be represented as

$$\dot{\mathbf{x}} = \boldsymbol{f}(\mathbf{x}) + [0, 0, 0, 0, 0, 1]^T u, \tag{5.31}$$

where $\mathbf{x} = [\varphi_1, \varphi_2, s, \dot{\varphi}_1, \dot{\varphi}_2, \dot{s}]^T$ is the state vector and s represents the displacement of the pendulum cart. We choose the acceleration of the pendulum cart as control input so that $u = \ddot{s}$. We refer to [7] for a derivation of the equations of motion and a technique to estimate parameters from experimental data.

We seek to perform the swing-up control of the double pendulum. However, the employed model is flawed, as it contains an additional term and an incorrect parameter:

$$\boldsymbol{f}_m(\mathbf{x}) = \boldsymbol{f}(\mathbf{x}) + [\sin(\varphi_1), 0, 0, 0, 0, 0]^T + [0, 0, 0, 0, 0, 0.95]^T u. \tag{5.32}$$

The discrepancy model is then

$$\delta\dot{\mathbf{x}} = \dot{\mathbf{x}} - \boldsymbol{f}_m(\mathbf{x}) = [-\sin(\varphi_1), 0, 0, 0, 0, 0.05u]^T, \tag{5.33}$$

which we seek to model.

First, we generate data for the identification using the imperfect model. Specifically, we design a feed-forward control input to swing up the double pendulum based on the imperfect model $\boldsymbol{f}_m(\mathbf{x})$. When this control signal is applied to the actual system, the observed behavior deviates from the pre-planned trajectory due to the model discrepancy. This difference is then used to form a sparse regression problem, as in Eq. (5.17), to identify the discrepancy $\delta\dot{\boldsymbol{x}}$. Finally, a new swing-up trajectory is designed using the hybrid model consisting of the imperfect model, augmented with the discrepancy model.

Results

Simulation results for the swing-up of the double pendulum are shown in Fig. 5.5. The feed-forward trajectory is determined as the solution to an optimization problem [366]. The simulation time step is $\Delta t = 0.001$, the prediction horizon is 2, and the total swing-up time is chosen to be $T = 2.7$. The weight matrices are $Q = \text{diag}[10, 10, 20, 1, 1, 0.1]$ and $R = 1$ for the state and input, respectively. It is demonstrated in Fig. 5.5 that SINDy correctly

identifies the model mismatch shown in Eq. (5.33) and that the new model mimics the actual dynamics of the system, which results in a successful swing-up of the pendulum.

5.3.4 Discrepancy Modeling with Modified SINDy

Modified SINDy can be easily integrated with the discrepancy modeling framework of SINDy. Suppose the known (theoretical) right-hand side dynamics in Eq. 2.1 is $\boldsymbol{g}(\mathbf{x})$ and let's assumes that the known model is not capable of modeling the data due to missing physics terms on the right-hand side. Thus there is a mismatch between the derivative $\dot{\mathbf{x}}$ and the known dynamics $\boldsymbol{g}(\mathbf{x})$. The discrepancy modeling approach tries to identify the missing dynamics $\boldsymbol{\Theta}(\mathbf{x})\boldsymbol{\Xi}$ such that

$$\dot{\mathbf{x}} = \boldsymbol{f}(\mathbf{x}) = \boldsymbol{g}(\mathbf{x}) + \boldsymbol{\Theta}(\mathbf{x})\boldsymbol{\Xi}. \tag{5.34}$$

To illustrate this process, consider a system $\dot{\mathbf{x}} = \boldsymbol{f}(\mathbf{x})$, whose model is given as

$$\begin{aligned} \dot{x} &= -10x + 10y + xy, \\ \dot{y} &= 28x - xz - y + 3z, \\ \dot{z} &= xy - 8/3z. \end{aligned} \tag{5.35}$$

Eq. (5.35) is simulated with the $x_0 = [5, 5, 25]$, $T = 30$, and $dt = 0.005$ to generate noise-free data. Training data is produced by adding 10% Gaussian noise in order to create the noisy measurement. Assume that the noisy measurement of Eq. (5.35) is given. Further assume that the dynamics is modified based on $\boldsymbol{g}(\mathbf{x})$, which is given by

$$\begin{aligned} \dot{x} &= -9.5x + 10.5y, \\ \dot{y} &= 27.6x - 1.1xz - 0.9y, \\ \dot{z} &= 1.05xy - 2.6z. \end{aligned} \tag{5.36}$$

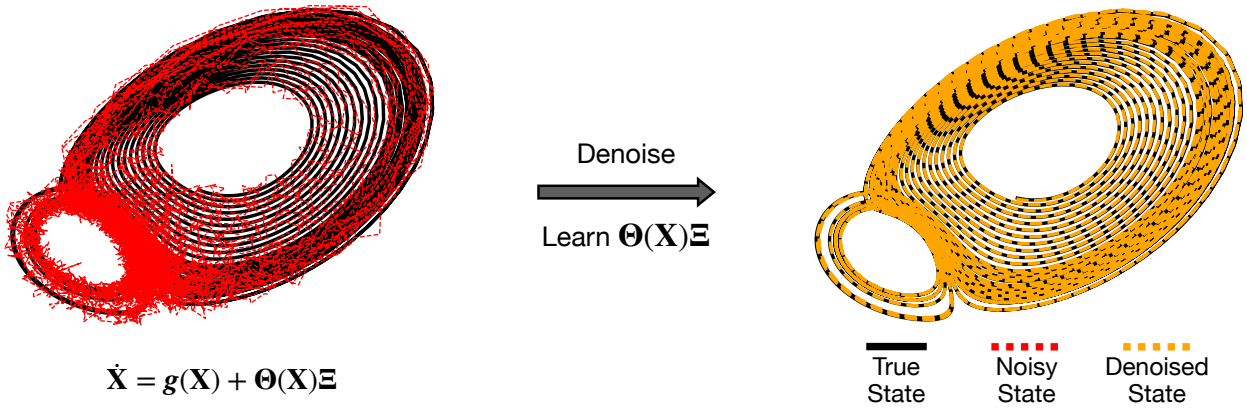


Figure 5.6: The modified SINDy can be easily integrated with the discrepancy learning method. The known structure $\boldsymbol{g}(\mathbf{X})$ can be utilized while we aim to learn the discrepancy and denoising the measurement.

The difference between the Eq. (5.35) and Eq. (5.36) will be the discrepancy model $\boldsymbol{\Theta}(\mathbf{x})\boldsymbol{\Xi}$ that modified SINDy identifies. Note that this prior information of the dynamics, $\boldsymbol{g}(\mathbf{x})$, can be constrained to exist in the modified SINDy library during the optimization process, and its parameters can be used as an initial guess of the true parameters. Thus, the only thing we have to learn is the missing dynamics. Fig. 5.6 illustrates this process and suggests that modified SINDy can be used to learn the discrepancy model when parts of the dynamics are already known.

5.4 Conclusion

In this Chapter, we present a data-driven framework to model discrepancies between observations and simplified or incorrect physical models. In particular, we leverage the sparse identification of nonlinear dynamics (SINDy) [180] algorithm to discover model discrepancies caused by parameter mismatch or model inadequacy. We demonstrate this approach on several systems, including the Van der Pol oscillator and experimental and numerical data from a double pendulum on an actuated cart. It is also shown that the modified SINDy can be integrated with a discrepancy modeling framework whereby prior

information of the dynamical model can be used to help identify the missing dynamics.

Our results suggest that a hybrid discrepancy modeling approach, involving a physics-based model and a data-driven correction, may have several benefits. First, we incorporate prior knowledge and enforce conservation laws and constraints that are notoriously challenging in data-driven approaches. In addition, focusing SINDy on the model mismatch is a much simpler task than trying to model all system dynamics at once, which would involve a large library of candidate terms and a potentially ill-conditioned inverse problem. This ill-conditioned problem often leads to the mis-estimation of parameters, such as the mass and length of the pendulum arms, which are accurately measured ahead of time. Instead, we are able to focus the data-driven effort on modeling the few terms and parameters that cause the discrepancy.

There are several future directions suggested by this work. First, it will be important to try out more variants of the SINDy algorithm in the discrepancy modeling framework. It will also be interesting to use our discrepancy models for the experimental swing-up control of the double pendulum on a cart, which is the subject of ongoing work.

Chapter 6

THE EXPERIMENTAL MULTI-ARM PENDULUM ON A CART: A BENCHMARK SYSTEM FOR CHAOS, LEARNING, AND CONTROL

Our work on data-driven modeling motivates us to build a benchmark experimental system that can facilitate the study of chaos, data-driven algorithms, and control algorithms. We developed a multi-link pendulum on the cart experimental system with those tasks in mind, and we will present a detailed tutorial on how to build it in this Chapter.

6.1 Introduction

In its simplest form, the single gravity pendulum constitutes a body suspended by a cord or rod that swings back and forth under the influence of gravity [367]. Investigations of this nonlinear system date at least to the seventeenth century and the work of Galileo Galelei [368–370], with derivations and explanations of the dynamics now commonplace in introductory mechanics courses. Perhaps the most well known application of the single pendulum is in the measurement of time, with Christiaan Huygens' proposed pendulum clock concept of 1656 being the most accurate time keeping device until the 1930s. The predictable oscillatory motion has also been used to infer the constant of gravitational acceleration, g [371]. As a mechanical benchmark system, the single pendulum has many notable variants: the single pendulum on a cart [372, 373], Foucault's pendulum [374, 375], Furuta's pendulum [376], the vertical take-off and landing (VTOL) single pendulum [147], the inertial wheel pendulum [377, 378], the spherical pendulum on a puck [379], and the inverted wheel pendulum [380].

In the eighteenth century, Daniel Bernoulli [381] advanced the study of the pendulum by introducing a second mass suspended by a cord or rod from the first mass, resulting

in the *double pendulum*[1] [381–383]. Although the gravity pendulum leads to predictable periodic motion, the double pendulum is a prototypical example of a chaotic system [384–387], requiring specialized numerical integration techniques such as symplectic [388] and variational [389] integrators. The double pendulum has played a central role in the historical development of dynamical systems, attracting the attention of early pioneers, such as Johann Bernoulli and D'Alembert [382]. Notable variants of the system include: the double pendulum on the cart [390], the rotary double pendulum [391–393], the "acrobot" [394], which is a double pendulum with actuation torque on the second arm, and the "pendubot" [395, 396], which is a double pendulum with actuation torque on the first arm. The double pendulum remains relevant in the study of nonlinear dynamics and has emerged as an important benchmark problem in system identification [13, 147, 177, 236, 397] and machine learning [6, 21, 147, 152, 236, 380, 398–400].

Beyond the double pendulum, one can continue adding "arms" (masses suspended by rods attached to the previous point masses) to form a chain resulting in the triple pendulum, and so on. The study of such pendulum systems have a long tradition in classical mechanics for a good reason: they are simple mechanical analogs that display much of the rich dynamical behavior observed in far more complex systems. Figure 6.1 illustrates the single, double, and triple pendulum.

The single, double, and triple pendulum are also widely used in the control community to develop and test new algorithms. The wide adoption of the multi-arm pendulum as a benchmark problem stems from the simple derivation of the equations of motion, the tunable complexity of behavior, and to the wide applicability in the physical sciences, including to robotics [401], engineering [402], and biology [403, 404]. In the case of the single pendulum, controllers have been designed to swing the pendulum up and/or stabilize it in the inverted position [147, 405, 406, 406–418]. Similar control methods have been developed to swing up the arms of the double and triple pendulums [7, 390, 393, 417,

[1]The double pendulum can also be referred to as a compound or linked pendulum.

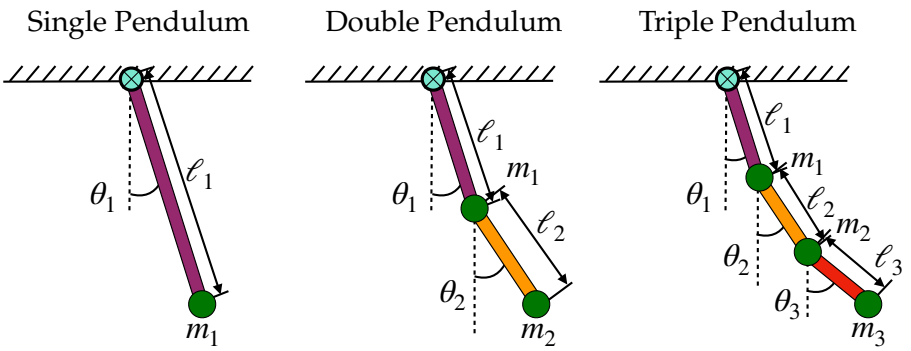


Figure 6.1: A schematic illustration of the single, double, and triple pendulum.

419–430], stabilize their arms in various unstable vertical positions [417, 419, 431–433], and perform time-periodic motion [391, 392, 396, 434–436]. Due to the chaotic nature of multi-armed pendulums, the sensitivity increases as more arms are added to the pendulum, thus making it an increasingly difficult control benchmark. The multi-arm pendulum is characterized by the existence of index-one saddle points (i.e., saddle points with exactly one unstable direction), that mediate the chaotic transport of trajectories in phase space; this provides an analog for several more complex systems, such as transport in the solar system [437–439] and chemical reaction kinetics [353].

With such interest in understanding and controlling the dynamics of pendulums, a number of researchers have build physical models to visualize and test their theoretical calculations; these experimental demonstrations include the single [372, 373, 375–378, 395, 397, 405, 407, 409, 410, 412, 416, 417, 440, 440–442], double [178, 384, 386, 390, 394–396, 417, 425, 443–446], and triple pendulum [428, 429, 431–433, 447, 448]. Among these physically-realized pendulums, it is the pendulum on a moving cart that has received the most attention. The general idea is to use the motion of the cart to control and stabilize the motion of the pendulum arms, with benchmark control problems typically focusing on forcing the arms into the unstable upright vertical position. Such a control problem is a proxy for more complex upright stabilization, including human beings standing using their feet as the pivot and applying small muscular adjustments to remain in the upright position. This control procedure is becoming increasingly important for the development of robots

that stand upright, and for personal transportation devices, such as self-balancing scooters and single-wheeled electric unicycles. Much like these physical balancing problems, the difficulty in controlling pendulums on a cart usually lies in the physical constraints of the system, such as the pendulum cart having a limited travel range and velocity. Other difficulties come from the specific hardware of the system, the control authority of the cart, and the placement of sensors for real-time measurements of the dynamics. The most common method to measure the rotational angle of each pendulum arm uses encoders and a motion capturing camera [14, 446], with variability in transmitting the collected data using hard wiring or wireless technology [429, 448]. In terms of the actuation of the cart, variants in the physical models use a rotational motor, a servo motor and belt drive, or a linear motor. All of these choices have a profound impact on the robustness of developed control methods.

In this paper we introduce the process and materials needed for constructing a high-performance, fully instrumented, multi-link pendulum on a cart. With regards to the variations on the physical model discussed above, we detail reasons for our choice of sensors, signal transmission, pendulum arm design, and actuation method. In particular, our pendulum on a cart uses an optical encoder and slip-ring to record and transmit the rotational angle of each arm, a linear motor for cart actuation, and Speedgoat and Simulink for the real-time control interface. The goal of this work is to provide a reference guide and tutorial for the construction of the pendulum on a cart system with a flexible design that can further be mounted for nonlinear dynamical demonstrations or easily altered to various other pendulum models to initiate new studies. Our contributions are as follows:

1. A detailed tutorial on how to build a multi-link pendulum on a cart system.
2. Open-source design files, including 3D CAD files and the Simulink files for data collection and control of the system.
3. Open access data sets of the pendulum system with its rotational angle and velocity recorded with and without control input.

These data sets could be of great benefit to the system identification, machine learning, and

artificial intelligence communities to test their techniques and algorithms. Finally, in the conclusion we introduce the concept of cloud experiments so that researchers worldwide can access our model and run experiments without having to build their own system.

We organize our work as follows: In Sec. 6.2, we present the major components of the multi-arm cart pendulum system. In Sec. 6.3, we describe the design and manufacturing procedure of the pendulum arm. In Sec. 6.4, we show the design process of the pendulum cart. In Sec. 6.5, the selection of the real-time control interface is presented. In Sec. 6.6, we discuss the electrical components of the system and illustrate the wiring specification of the entire system. Finally, Sec. 6.7 summarizes the work and explores the idea of cloud experiments. Sec. 6.7 also includes the discussion of software setup, operation and safety, and parameter estimation of the pendulum arms.

6.2 Overview of the System

In this section we introduce the main components, design, and closed-loop control of the multi-link pendulum on the cart system. The system can be used to gather experimental data of a single, double and triple pendulum's oscillatory motion, and can be used to validate different control laws and system identification algorithms. An overview of the system can be seen in Fig. 6.2.

The proposed system consists of four major components: 1) the servo drive (motor drive), which provides proper current to the linear motor to move it according to the desired speed; 2) the linear motor with a $1\mu m$ resolution magnetic encoder, which provides actuation forces to the pendulum; 3) the real-time system, which controls the motion of the pendulum cart and pendulum arm; and 4) the pendulum arm, which can rotate freely around its joint. The pendulum arm has an optical encoder with 10000 counts per revolution (CPR) to measure its angular position. The four main components are mounted on a frame, as shown in Fig. 6.2.

In order to form the closed-loop control, the measurements from the pendulum arm and linear motor are sent to the Speedgoat real-time system. The real-time system then

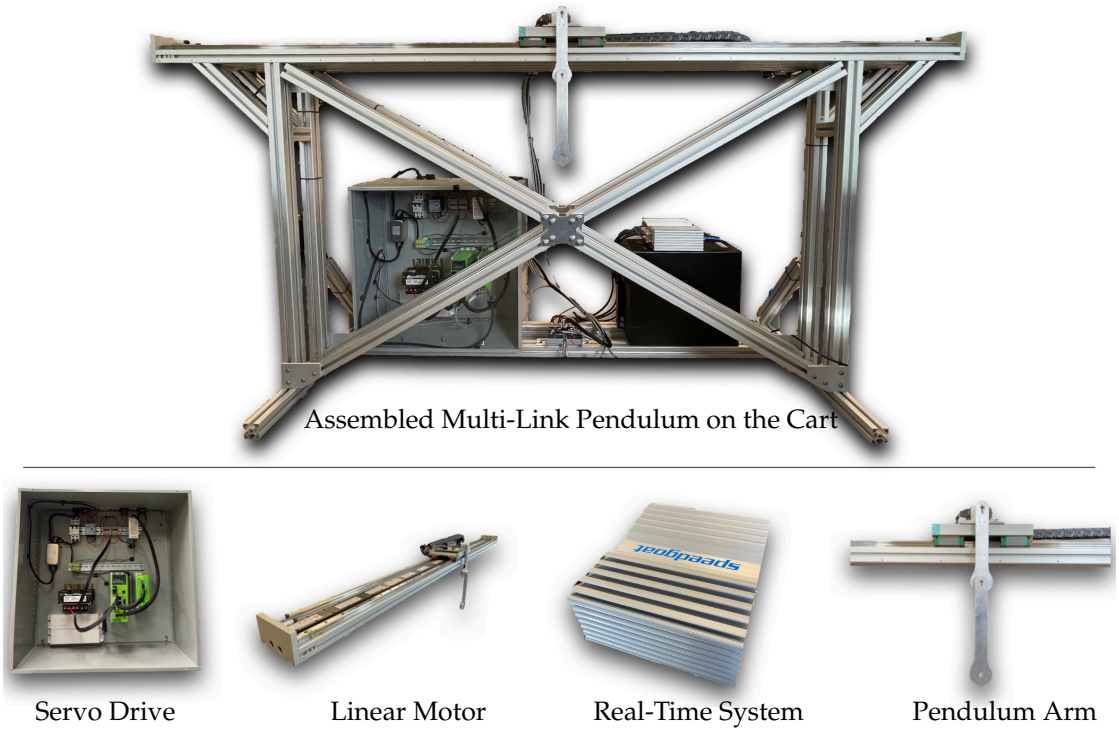


Figure 6.2: Assembled multi-link pendulum on the cart system. The major components of the system are the servo drive, linear motor, real-time control system, and pendulum arm on the cart.

runs the user defined controller and calculates the control action needed for the next time step given the control objective. The corresponding control value is then converted to an analog voltage output, and this voltage is sent to the servo drive. In velocity mode, the servo drive measures the voltage of the analog signal generated by the real-time system, and determines the desired velocity of the linear motor. To achieve the desired velocity required by the user, the servo drive uses encoder measurements from the linear motor to calculate its position and velocity. By comparing the actual and the user defined velocity of the linear motor, the servo drive internally[2] calculates the currents needed to achieve the target velocity. The closed loop control diagram is illustrated in Fig. 6.3.

Our pendulum design enables simple manufacturing and reliable and accurate op-

[2]To the best of our knowledge, the servo drive uses a PID controller to achieve the target speed set by the user.

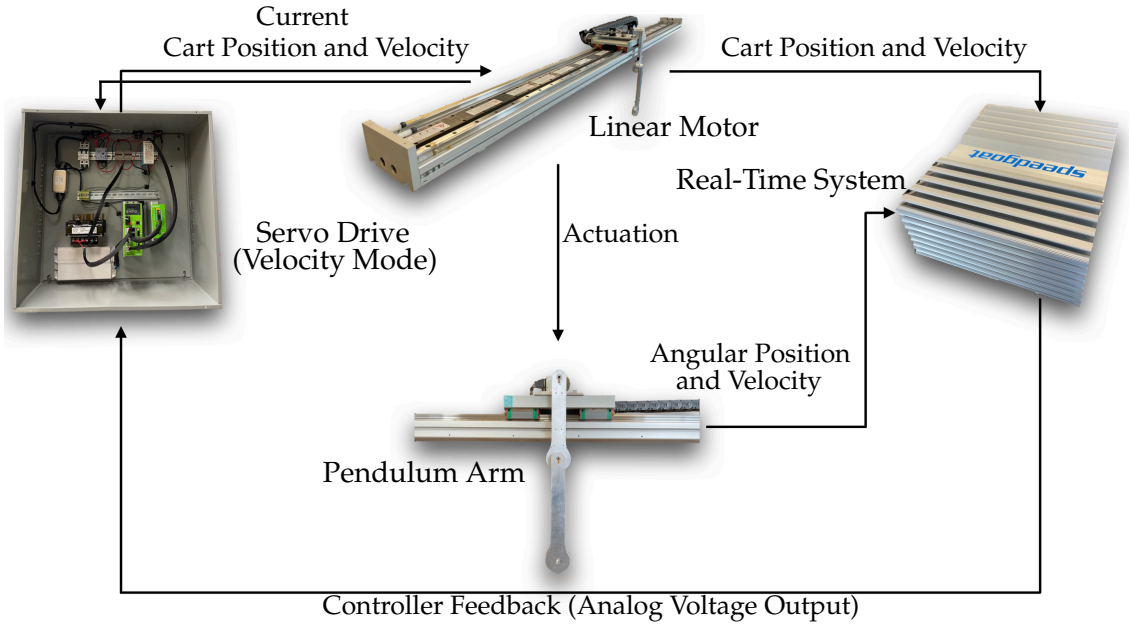


Figure 6.3: Closed loop control diagram of the proposed system. The linear motor and pendulum arm sends measurement data to the real-time system, which is used to calculate the corresponding control action needed to achieve the control object, such as stabilization, swing-up, stabilizing periodic orbit, etc. After the control command is received by the servo drive as an analog signal, the servo drive controls the linear motor to achieve the desired motion.

eration. It offers several advantages compared to alternative designs: 1) A linear motor overcomes the backlash that can occur in belt-driven cart systems, which poses a challenge for multi-link pendulum control. 2) Using a slip-ring to transfer the electrical signal in the rotating pendulum arm avoids latency that may occur in pendulum designs with wireless transmission. Moreover, using slip-rings avoids adding a battery to the pendulum arm and enables light weight designs [429, 448]. However, using slip-rings results in a more complicated pendulum arm design, which is more difficult to machine. Also, due to the limited number of channels the slip-ring provides (5 or 8 signal wires), it is difficult to add a gyro sensor to measure the acceleration of the pendulum arm. 3) The Speedgoat baseline machine used as our real-time control system fully supports the Simulink Real-Time software, which simplifies the controller validation and testing. In case the system

is designed to only study a single pendulum, the Speedgoat machine can be replaced with a less expensive solution, discussed in Appendix. C.1. 4) The same system can be used to study both controlled and uncontrolled behavior, and the user can add/detach pendulum arms to study single, double, or triple pendulums using one system. 5) The pendulum can be detached from the linear motor, and the system can be used to perform other experiments, such as oscillation studies. 6) Different pendulum arm designs can be installed and tested. Thus, the resulting system has great flexibility for future extensions.

6.3 Pendulum Arm

The main component of the cart-pendulum system is the pendulum arm. The most simple design consists of an off-the-shelf rod with a mass attached to its end. This design is widely used for single and double pendulums due to its simplicity, but it limits the integration of sensors to measure angular position. For the triple pendulum, it is more common to machine custom aluminum pendulum arms with integrated sensors that can record the rotational angle of the pendulum. In this section, we detail our multi-link pendulum arm design based on custom machined parts with integrated sensors. We introduce the pendulum arm design and assembly, and refer to Appendix. C.2 for a detailed description of the design and manufacturing of the arm.

6.3.1 Design

An overview of our pendulum arm design is illustrated in Fig. 6.4. The main components of the pendulum arm are: 1) pendulum body; 2) shaft; 3) bearing plate; and 4) 3D printed protection case. The overall structure of the pendulum arm is determined by how it transmits the rotational information measured by the encoders. In our design, a slip-ring sends the encoders' electrical signals to the real-time system. The advantage of the slip-ring design is the low latency in the signal transmission compared to vision-based and wireless communication systems. Also, no additional computational resources are needed

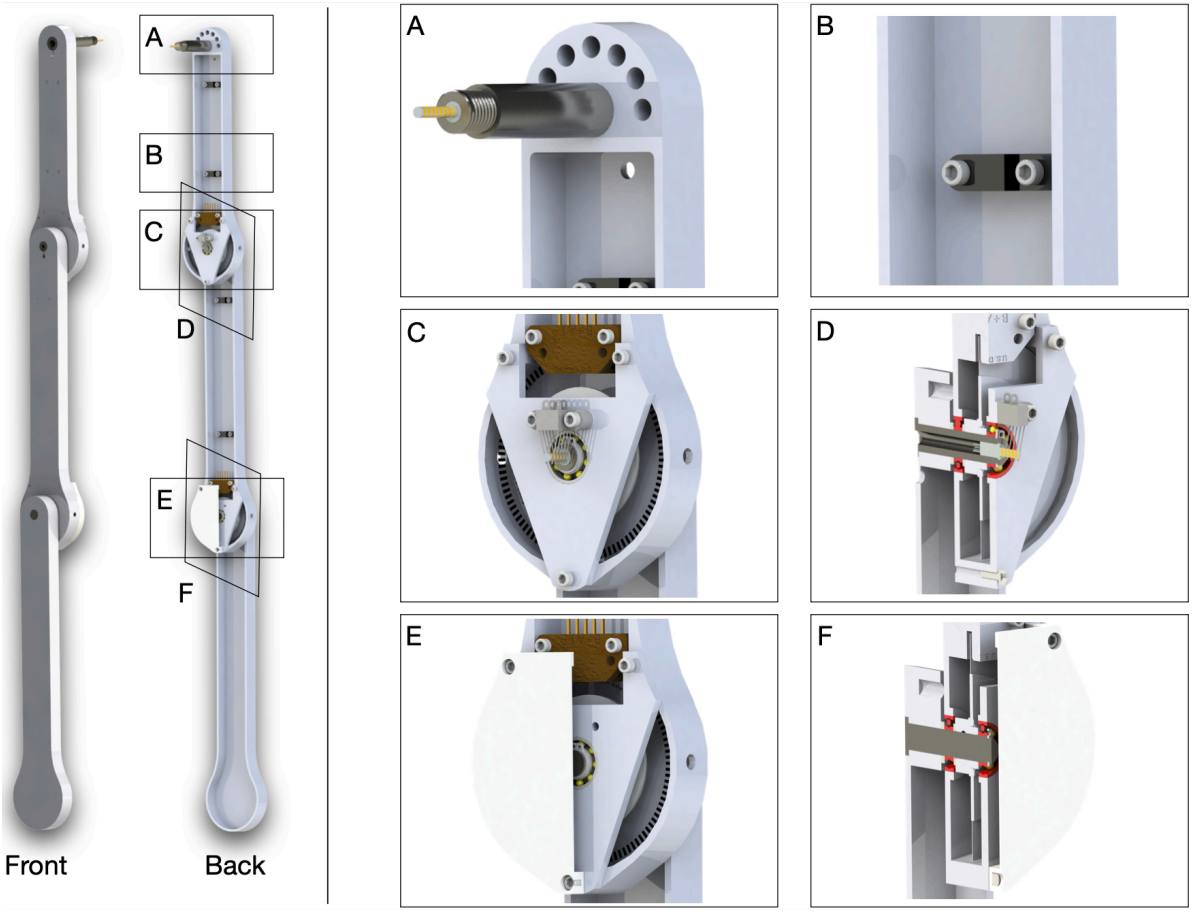


Figure 6.4: Design of the multi-link pendulum arm. The main components of the assembled pendulum arm are: 1) the pendulum arm body, which is used to install different parts of the pendulum arm, such as sensors, wires, slip-ring, etc.; 2) the pendulum shaft, which is mainly used to support the rotational movement of the pendulum arm; 3) the bearing plate, which is used to secure the pendulum shaft and connect two pendulum arms; and 4) the protection case, which is used to enclose the sensors and slip-ring and prevent them from being damaged.

to determine the rotational angle of the pendulum, compared to vision-based tracking systems. This characteristic is particularly beneficial for achieving high sampling rates. One drawback of the slip-ring design is the additional friction on the contact between the slip-ring shaft and brush block. This can be minimized by using a miniature slip-ring and slip-ring brush with gold contact surfaces, which also reduce the electrical noise during the

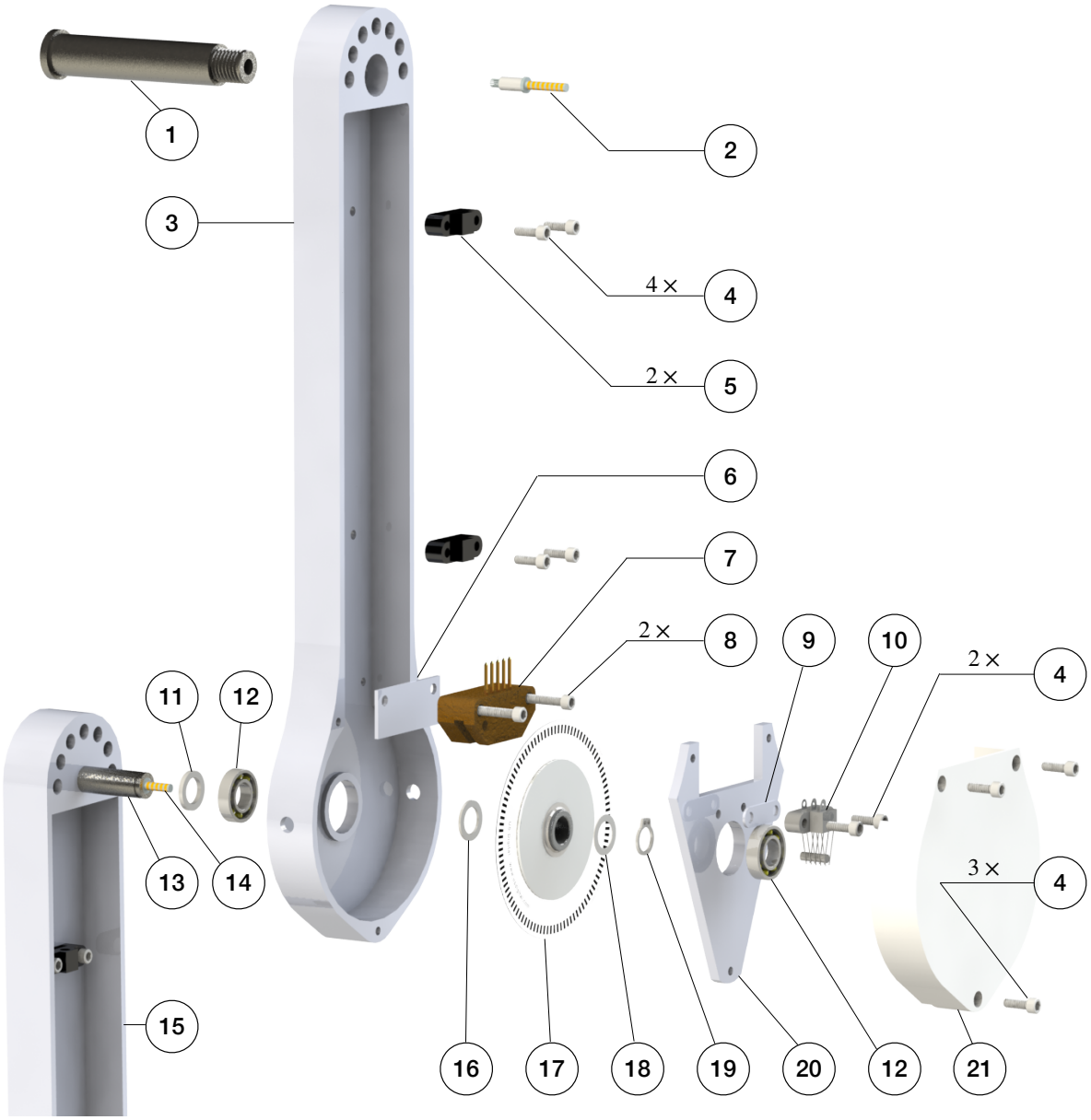


Figure 6.5: Components and assembly of the first and second pendulum arm.

rotational movement of the pendulum arm. An additional challenge of using slip-rings is the requirement for precision machining of the pendulum shaft and also the fixed number of channels to transmit signals. This may reduce the flexibility of the setup if new sensors (e.g. an inertial measurement unit (IMU) sensor) are required for future experiments.

In our design, the shafts of the first and second pendulum arms are hollow to accommodate the slip-ring and the connections to the sensors. The slip-ring wires are clipped to the pendulum arm to prevent twining of the cables. Moreover, two holes on the side of the pendulum arm facilitate the installation of the encoder disk on the pendulum shaft. A stair case shoulder properly secures the bearing that is installed on the first and second pendulum arm. Ceramic bearings are used to minimize the friction during the rotational movement. The advantage of ceramic bearings is that they operate without lubrication. Two bearings are used to fully support the rotational movement of the pendulum arm. To avoid the pendulum shaft sliding out of the pendulum arm during operation, external retaining rings are used to secure the pendulum shaft. The main components of the pendulum arm are manufactured using CNC milling and turning. Further details are provided in Appendix. C.2.

6.3.2 Assembly

The assembly of the pendulum arms is shown in Fig. 6.5 and Fig. 6.6. The pendulum body and bearing plate are assembled first. In order to assemble the pendulum arm body, the following steps are performed: 1) the bearing (12) is installed to the pendulum arm body (3, 15) (the transitional fit between the bearing and bearing hole should facilitate the assembly). To make sure the bearing stays in place, a thin layer of Loctite is applied to the outer ream of the bearing before it is push into the bearing hole. Once the bearing is installed, a paint tape is used to cover the bearing to prevent dust entering the assembly. 2) The pendulum shaft (1, 13, 22) is press fitted into the pendulum arm body (3, 15, 23). To make sure the installation is smooth, it is recommended to lubricate the pendulum shaft and shaft hole before the press fit. 3) The slip-ring (2, 14) is then installed onto the pendulum shaft (1, 13). Those steps complete the assembly of the individual pendulum arm. The assembly of the bearing plate follows a similar process. The outer ream of the bearing (12) is applied with a thin layer of Loctite, then pressed into the bearing plate (20).

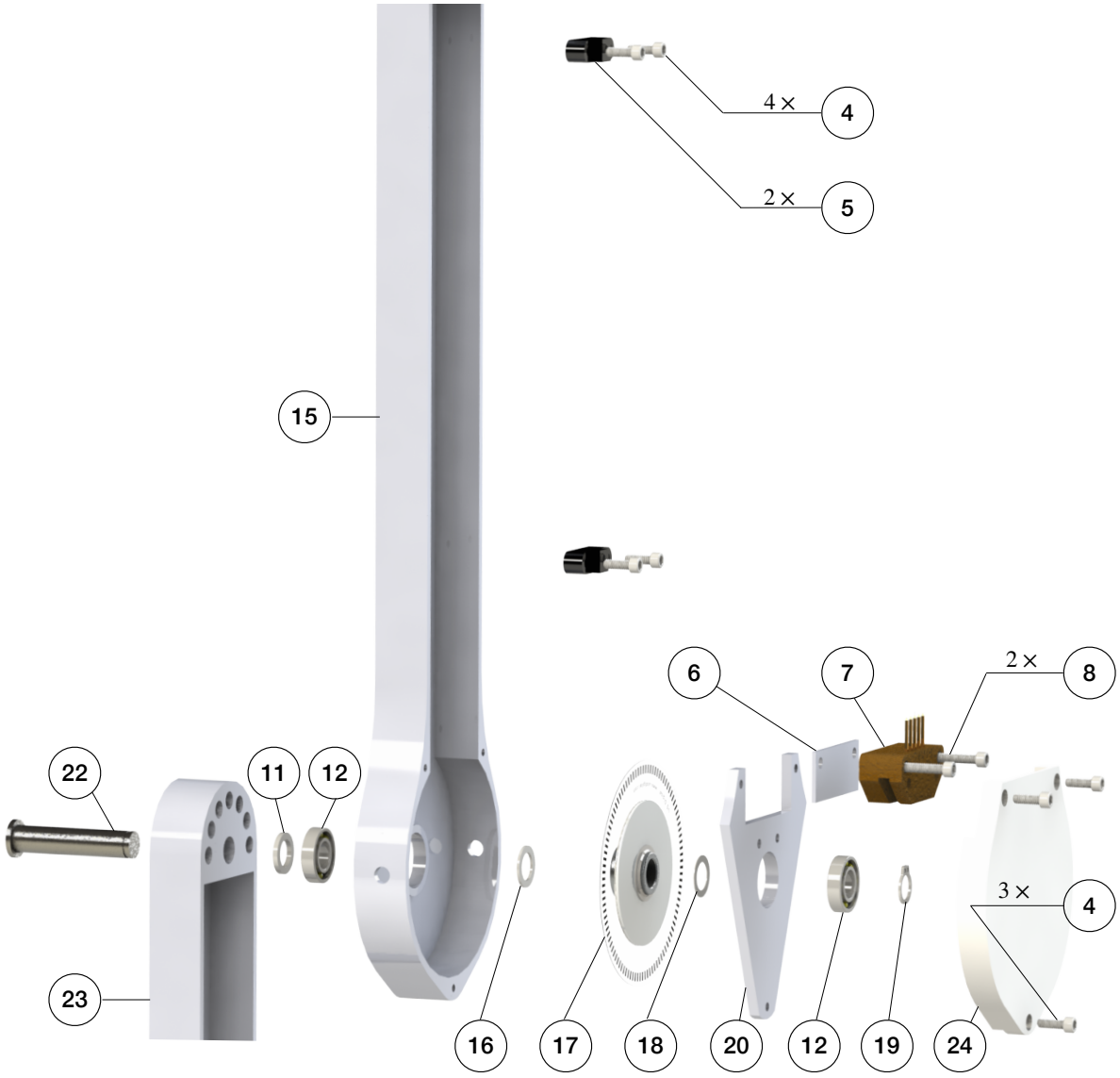


Figure 6.6: Components and assembly of the second and third pendulum arm.

This step should also be straightforward, since the fit tolerance between the bearing and bearing plate is a transitional fit.

To assemble the double pendulum (first and second arm), the following steps are performed: 1) the shim (11) is slid onto the shaft of the second arm (13). Next, the pendulum shaft is slid into the bearing of the first arm (12) until the shim (11) contacts

the inner ring of the bearing. 2) The shim (16), encoder disk (17), and another shim (18) are slid onto the second arm's shaft (13). Then, the assembled bearing plate is slid onto the shaft (13) as well. Finally, the external retaining ring is clipped onto the shaft (13). This should hold everything together while still allowing adjustments, since the bearing plate is not screwed yet. 3) The 3D printed shim (6) and the encoder reader (7) are pushed onto the desired place by aligning the holes. Then, the screws (8) are positioned. 4) The 3D printed shim (9) and slip-ring brush block (10) are installed by aligning the holes on the bearing plate, and the screws (8) are tightened. After this step, the contact between the slip-ring and slip-ring brush block should be carefully observed. The slip-ring brush should be centered to its corresponding channel. 5) Next, the protection case is installed by aligning the holes on the pendulum arm body, bearing plate, and protection case. The screws (4) are mounted to fasten the assembly. 6) Finally, the wire clipper (5) is installed onto the pendulum arm (3) by aligning the holes, and the screws (4) are tightened. The above steps finish the assembly of the first and second pendulum arm. Fig. 6.6 shows the assembly of the triple pendulum arm, following similar steps as for the assembly of the double pendulum arm. The difference though is that the third arm's shaft does not have a slip-ring installed. In Appendix. C.2 the detailed steps to assemble the triple pendulum arm are introduced.

6.4 Pendulum Cart

In this section we introduce the design and assembly process of the pendulum cart. We first discuss the motor type selection and sizing, and then introduce the design and assembly process of the pendulum cart and the bearing house that is used to connect the pendulum arm to the motor, as shown in Fig. 6.4. The detailed design and manufacturing of the cart and linear motor support frame is introduced Appendix. C.3 and C.4.

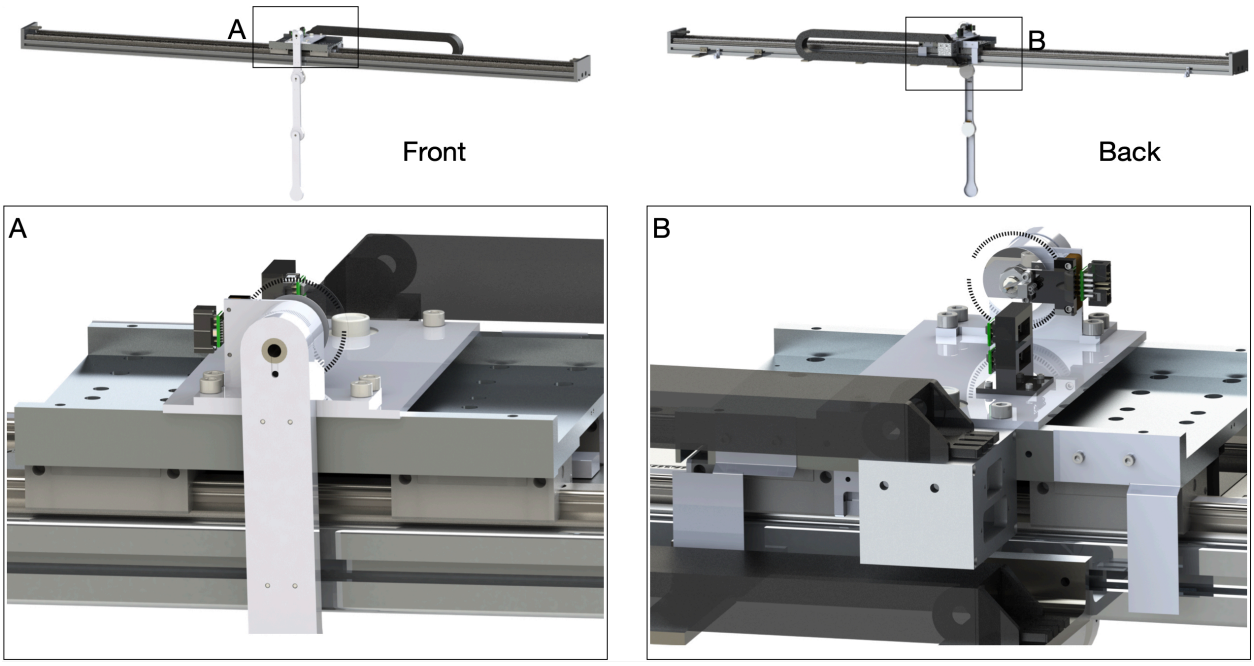


Figure 6.7: Overall assembly of the pendulum arm and pendulum cart. The main components of the pendulum cart are an aluminum plate and a bearing house.

6.4.1 Design

The pendulum cart has two main functionalities: 1) mounting and supporting the pendulum arm; and 2) providing the actuation to the first pendulum arm to control the system dynamics. The first step in the design of the pendulum cart is the selection and sizing of the motor. Once the motor type and size are selected, a mechanism has to be designed to connect the pendulum arm and motor. Fig. 6.7 illustrates the designed pendulum cart with the assembled pendulum arm.

The actuation of the pendulum arm is provided by a linear motor. The advantage of using a linear motor is that it does not have backslash issues, which frequently happen in belt-drive type servo motors. This allows accurate control of sensitive maneuvers, such as the swing-up of the double and triple pendulum. Once the type of linear motor is determined, the next step is to size the linear motor. First, the desired maximum speed and

acceleration of the cart is defined. To determine the maximum speed and acceleration, we look at the pendulum cart's desired motion profile. We take the feed-forward trajectory [7] that is needed to swing up the double or triple pendulum. Using this desired motion profile, we determine the peak force and velocity required and use this information to size the linear motor. Here, we determine the pendulum cart's top speed and acceleration to be $5m/s$ and $20m/s^2$. The pendulum arm we designed in Fig. 6.4 weights less than $0.5kg$ and the mass of the linear motor stage is $5kg$. Thus, the total continuous force provided by the linear motor should be around $110N$. We choose a HIWIN linear motor system [3] LMX1K-SA12-1-2000-PGS1-V103+HS. This linear motor can provide a peak force of $579N$ with a peak current of $12.7A_{rms}$ and a continuous force of $205N$ with a continuous current of $4.2A_{rms}$. This motor is powerful enough to provide the desired acceleration and speed. Moreover, the effective stroke of the linear motor is $2m$ with a magnetic incremental encoder of $1\mu m$ resolution.

Once the linear motor has been selected, the next step is to design the connection between the pendulum arm and the linear motor stage. As illustrated in Fig. 6.7 (A) and (B), a bearing housing is needed to provide support for the first pendulum arm shaft and to secure it so that the pendulum arm can perform free swing. Moreover, an aluminum plate is machined so that the bearing housing can be connected with the linear motor stage. The detailed design of the bearing house is introduced in Appendix. C.3. The final components we design for the pendulum cart are the limit switch plates, as shown in Fig. 6.7 (B). Fig. 6.7 (B) shows two limit switch plates installed on the back side of the linear motor stage. They are responsible to block the laser limit switch when the pendulum cart moves to the edge of the linear rail. Same as the pendulum arm, the main components of the pendulum cart are manufactured using CNC milling. The details of the design and manufacturing are provided in Appendix. C.3.

[3]See Appendix A of the HIWIN linear motor system manual document "Linear Motor System(EN).pdf"

6.4.2 Assembly

The assembly of the cart plate, bearing housing, pendulum arm, and the linear motor stage is illustrated in Fig. 6.8. First, the bearing housing is assembled, which includes two steps: 1) a 3D printed spacer (32) is press-fitted into the bearing hole (33). 2) A thin layer of Loctite is applied to the outer ream of the bearing (31). Then, the bearing is installed on the backside of the bearing house. Next, the same process is repeated to install the bearing on the front side. It is important to avoid excess Loctite to enter the bearing, which may damage it. Once finished, the bearing housing is left for 24 hours to ensure the bearing is securely installed. This process can be seen in Fig. 6.8 (B). Next, the cart plate is assembled. First, the circular level indicator (28) is installed to the cart plate (29). To secure it, a thin layer of Loctite is applied to the outer ream of the circular level. Next, the 3D printed cable management tool (25) is screwed (4) to the cart plate (29), and the differential driver (26) is installed on the cable management tool (25). Then, the cart plate (29) is mounted to the linear motor stage (30) using 4 screws (27). This completes the installation of the cart plate to the linear motor, as shown in Fig. 6.8 (A).

The installation of the pendulum arm to the bearing housing is shown in Fig. 6.8. 1) The spring steel ring shim (34) is slid onto the first pendulum arm shaft (1). Then the shaft (1) is slid into the bearing (31) until the shim contacts the inner ring of the bearing. 2) A spacing shim (35) is placed onto the shaft until it contacts the inner ring of the bearing (31). The encoder disk (36) is installed onto the shaft, which allows the measurement of the first arm's rotational angle. 3) A nut (37) is tightened using the thread on the first pendulum arm. This allows the application of axial force to the bearing housing, which helps to reduce the oscillation of the pendulum arm on the axial direction. The nut should not be tightened too much, to prevent damage of the bearing and to reduce the friction between the bearing ball and bearing case. 4) The encoder reader (7) is placed so that its holes are aligned with the holes on the bearing housing. Next, the holes are aligned on the 3D printed slip-ring brush base (38) with the holes on the bearing housing. Once the holes are

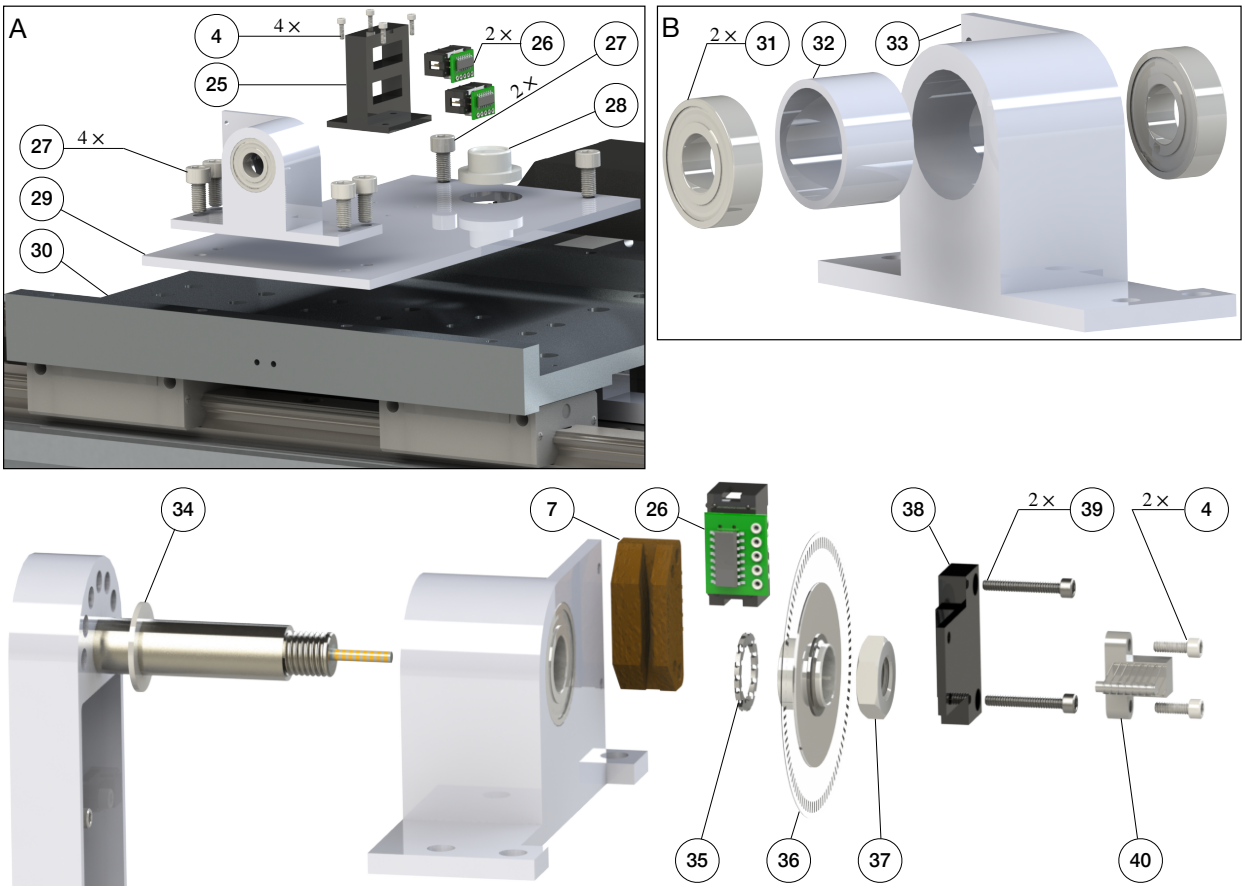


Figure 6.8: Assembly process of the pendulum cart, pendulum arm and bearing housing. (A) Installation of the bearing housing, cart plate and linear motor stage. (B) Assembly of the ceramic bearing and the bearing housing. (Bottom) Installation of the pendulum arm and bearing housing.

aligned, the bearing housing, encoder reader, and 3D printed slip-ring brush block base are mounted with screw (39). 5) The differential driver (26) is installed onto the encoder reader (7). 6) The slip-ring brush block (40) is installed onto the 3D printed base (38) with screw (4). It is important to make sure that each brush is centered with the corresponding channel on the slip-ring (2). The above steps complete the assembly of the pendulum arm and the linear motor.

The assembly of the limit switch plate (41, 43) and the linear motor stage (30) is shown

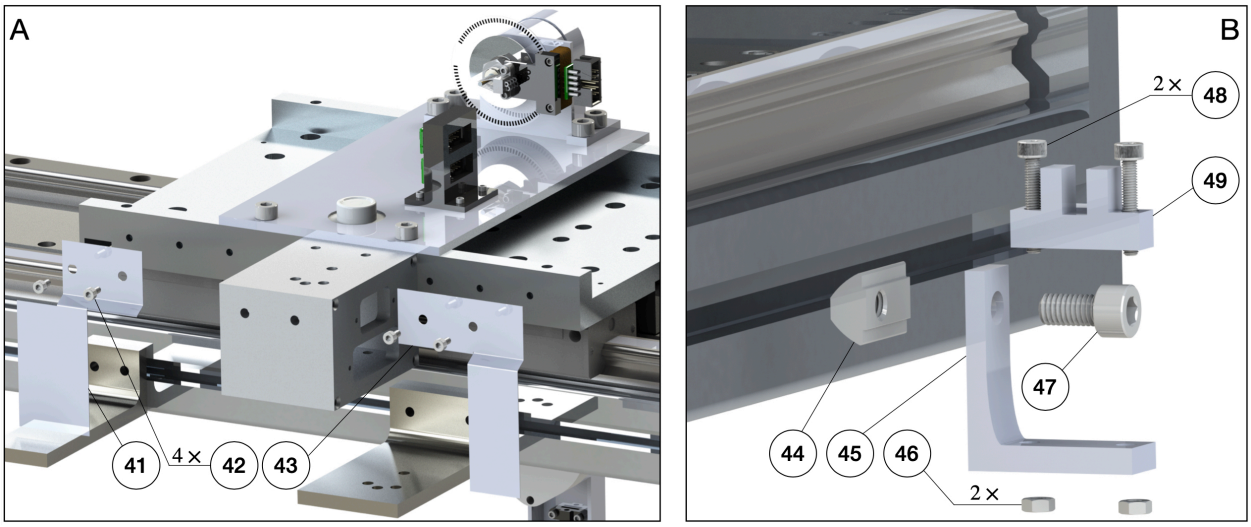


Figure 6.9: Assembly of the limit switch plate and limit switches.

in Fig. 6.9 (A). The left and right limit switches (49) are connected to the linear motor frame with the 3D printed limit switch base (45). The 3D printed base (45) is first connected with the linear motor frame using drop in T-slotted framing fasteners (44) and screws (47). Next, the limit switch (49) is connected with the 3D printed base using screws (48) and nuts (46). The detailed assembly process is illustrated in Fig. 6.9 (B).

6.5 Real-Time System

Sec. 6.3 and 6.4 introduce the major hardware components of the pendulum on the cart system. In this section, we introduce the real-time system that is responsible for: 1) processing the signal sent by the pendulum arm, linear motor, and other sensors; and 2) using the received signal to determine the control action in real-time based on the user program. The major components of the real-time system include: 1) baseline real-time target machine by Speedgoat. 2) User-selected I/O modules, including modules installed inside the real-time system, terminal boards connected to the sensors, and cables connecting the terminal boards and the real-time system. 3) Computer used to program the controller using Simulink Real-Time. 4) Other components such as the power cord,

power adapter, Ethernet cable, software drivers, etc. Fig. 6.10 (A) illustrates the overall Real-Time system. We use the Speedgoat machine with Simulink Real-Time for several reasons: 1) Matlab and Simulink have a wide application in both industry and academia. 2) Speedgoat and Simulink Real-Time simplify rapid prototyping and hardware in the loop control. 3) The Speedgoat machine has a responsive customer service that helps users to solve their software and hardware problems while using the Real-Time machine. One drawback of the system is that it is comparably expensive. In Appendix. C.1, we introduce an alternative custom-made Real-Time system using the National Instrument Data Acquisition (DAQ) board with Simulink Desktop Real-Time. Other alternatives of the Real-Time system include dSPACE, Typhoon HIL, National Instrument, and others [449].

6.5.1 System Choice and Specifications

The number and types of I/O channels and the desired sampling frequency determines which specific Speedgoat Real-Time system can be used. In our design, the Real-Time system needs to read at least four quadrature differential encoder signals (three from the pendulum arm and one from the linear motor). We need to have at least two digital input channels to read the limit switch signal, one digital output channel to enable/disable the linear motor drive, and one analog output to control the linear motor in velocity mode. The IO-191-EDU-Baseline as our Digital and Analog I/O module meets these requirements. The IO-191-EDU-Baseline is a 16-bit analog I/O module with eight single-ended or four differential sequential sampling analog inputs. The supported voltage ranges of the analog inputs are $\pm 0.64V$, $\pm 1.28V$, $\pm 2.56V$, $\pm 5.12V$, $\pm 10.24V$, $\pm 12.288V$, $\pm 20.48V$, and $\pm 24.576V$. Moreover, it has four single-ended analog outputs with supported voltage ranges of $\pm 10V$, $\pm 5V$, $\pm 2.5V$, $0 - 10V$, or $0 - 5V$. Both the input and output range of the analog signal is software configurable. Finally, it also has 16 x general-purpose digital TTL I/O lines. Thus, the module meets our requirements for digital and analog I/O capability. More details can be found in the IO-191-EDU-Baseline manual. As for the encoder reading, we select the IO-392-Baseline configurable FPGA-based I/O module with 50k Artix 7 FPGA and

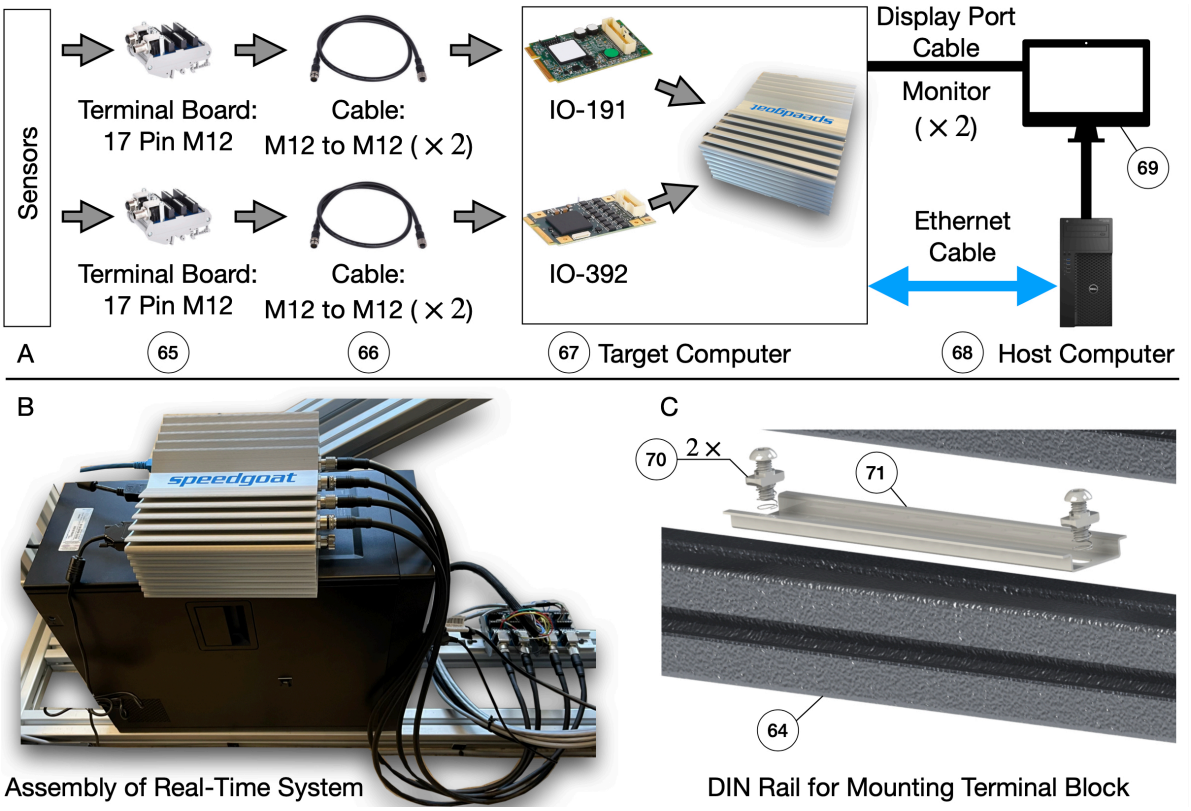


Figure 6.10: Assembly of the real-time system. (A) rough connection illustration of the target system. The terminal block is used to connect with the sensors. A cable is used to send the signals to the target system. The running status of the target machine is shown on the monitor and the target system and host computer is connected using an Ethernet cable which allows the update of the real-time program to be executed. (B) Assembly of the real-time system. The target machine is place on top of the host machine while the host computer is place on top the system frame. (C) Assembly of the DIN rail that is used to mount the terminal block of the real-time system.

13x RS422 digital I/O lines. This module can read four quadrature differential encoder signals while providing a $5V$ power supply to the encoder sensor. The Pin-Out map for this module while using the driver IO-392-QAD4RS422 can be seen in Table. C.1. More details can be found in the user manual [450].

After selecting the I/O modules, the next step is to determine which specific Speedgoat system is used. This is mainly determined by the sampling frequency, which depends on the user-specific program/algorithm. The desired sampling of the rotational information

of the pendulum arm is $5kHz$ when no control is applied. When stabilizing the single, double and triple pendulum, the sampling rate should be at least $1kHz$ (using LQR with Kalman filter). According to the user manual of the Baseline Speedgoat system, the I/O latency for reading and sending signals is around $31.5\mu s$ when using IO-191-EDU-Baseline and IO-392-Baseline. For the baseline machine, the algorithmic calculation takes about $56\mu s$ when running a Simulink model with 1550 blocks and 250 continuous states (equals to 25 Simulink benchmark model F14). Thus, the total latency time is around $87.5\mu s$, which theoretically enables a sampling rate of $11.4kHz$, which meets our requirements. Therefore, we choose a baseline real-time target machine by Speedgoat as our real-time controller. After testing, we found a maximum sampling rate of around $12.5kHz$ for pure data collection, and $5kHz$ during double/triple pendulum stabilization (using a time-varying LQR controller and a Kalman filter). The sampling rate is problem-specific and using a highly optimized code can further increase the maximum sampling rate. Finally, the host machine used for developing the control law has an Intel(R) Core(TM) i7-3770 CPU with 3.40GHz frequency and 32GB of RAM. The host machine runs on Windows 10 Pro with Matlab 2021b.

6.5.2 Assembly

The assembly of the real-time system is straightforward, as shown in Fig.6.10 (B). The host computer (68) is directly placed on the aluminum extrusion (64), while the target computer (67) is placed on top of the host computer. The Speedgoat target machine can be mounted on the system frame following the steps specified in the user manual of the real-time system. By directly placing the host and target machine on the system frame, the center of mass of the whole system is lowered, making the system frame more stable. A DIN rail (71) is mounted on the system frame (64) using a drop-in fastener. This DIN-rail allows the mounting of the terminal block (65). The bill of materials of the real-time system can be found in Table. C.8 in Appendix. C.10.

6.6 Electrical System

In this section we introduce the major electrical components of the experimental setup. We introduce the functionality of the components and illustrate the wiring specification of the entire system. The electrical system of the whole setup can be divided into two parts: 1) Linear motor power supply. 2) Pendulum arm sensor system. In the following, we introduce each part separately and illustrate the entire wiring diagram of the electrical system. WARNING: The wiring diagram of the system is provided as a reference and it has not been examined by an electrician. To the best of the author's knowledge, the provided wiring diagram is safe to be used but we do NOT GUARANTEE its safety.

6.6.1 Electrical Component of Pendulum Arm

The electrical component of the pendulum arm mainly consists of an encoder reader (7), slip-ring (2, 15), slip-ring brush block (10, 40), differential driver (25), and differential driver cable (73). Fig.6.11 (A) illustrates all those components. The encoder reader's single-ended A, B, and C/Index channel is transferred into the differential signal using differential drivers to improve the noise robustness of the encoder measurements. Using a differential signal helps to avoid the effect of electrical noise generated by the linear motor. It is generally recommended to use a differential signal whenever possible. Three US Digital CA-C10-SH-NC 10 feet cables are used (differential driver cable) to transfer the differential signals to the target computer. One end of the differential driver cable is connected to a 10-pin female standard (non-latching) connector. This connector is then inserted into the differential driver. The other end of the differential driver cable is unterminated. Thus, it is stripped and inserted into the corresponding channels on the real-time system terminal block (65). This process completes the connection of the differential driver and target system. Last, the differential driver and encoder reader are connected, as shown in Fig.6.11 (B).

The first pendulum arm's encoder reader (measuring the arm's rotational angle) is

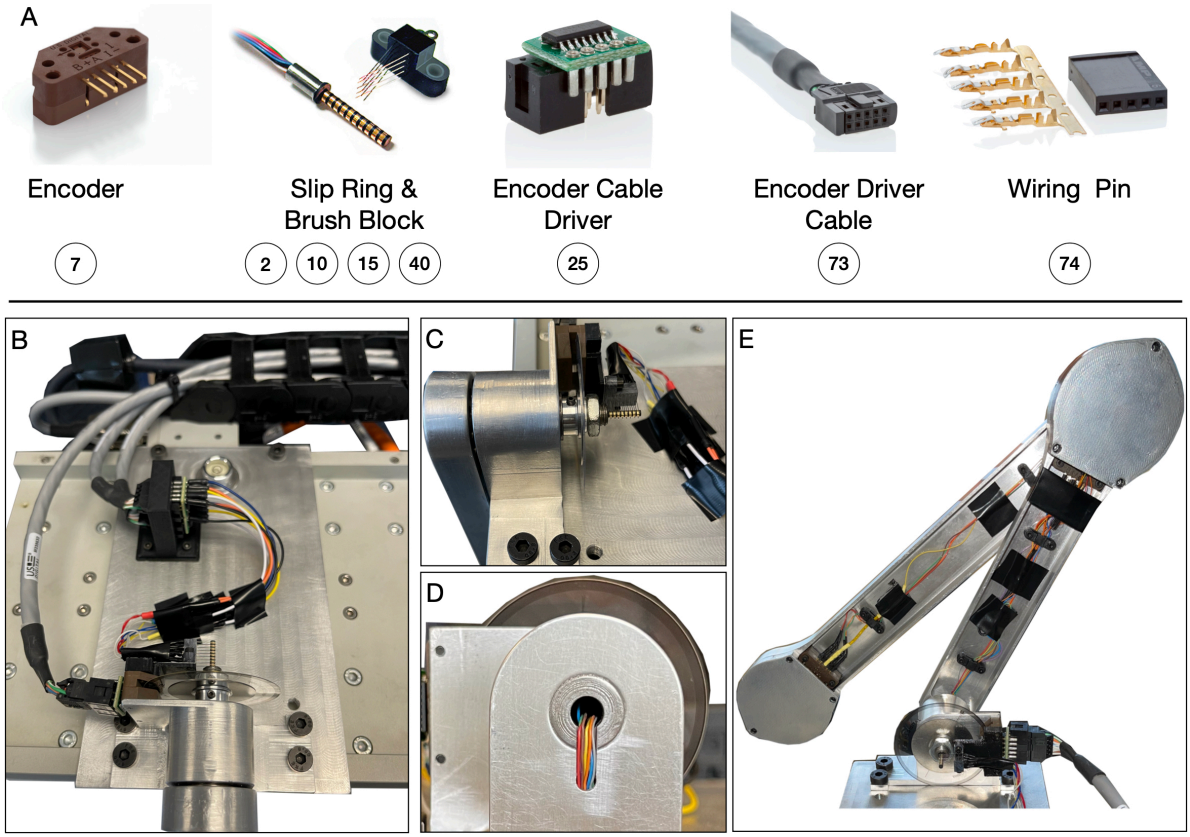


Figure 6.11: Electrical wiring and assembly of the pendulum arm. (A) Major electrical components of the pendulum arm. (B) connection method of differential driver and encoder reader. (C) to (E) wiring of the slip-ring.

directly connected to the differential driver, since the encoder reader is mounted on the bearing housing. This mounting position simplifies the connection of the differential driver to the encoder reader. However, the encoder reader and differential driver connection on the second and third arm are located inside the pendulum arm. This mounting position prohibits the direct connection of the driver and encoder reader. Therefore, two slip-rings are used that connect the encoder reader inside the first and second pendulum arm, as shown in Fig. 6.11 (E). This allows an indirect connection of the differential driver and encoder reader through the slip-ring brush block, which conducts the encoder reader signal out of the pendulum arm. Next, the breadboard jumper cable connects the slip-ring

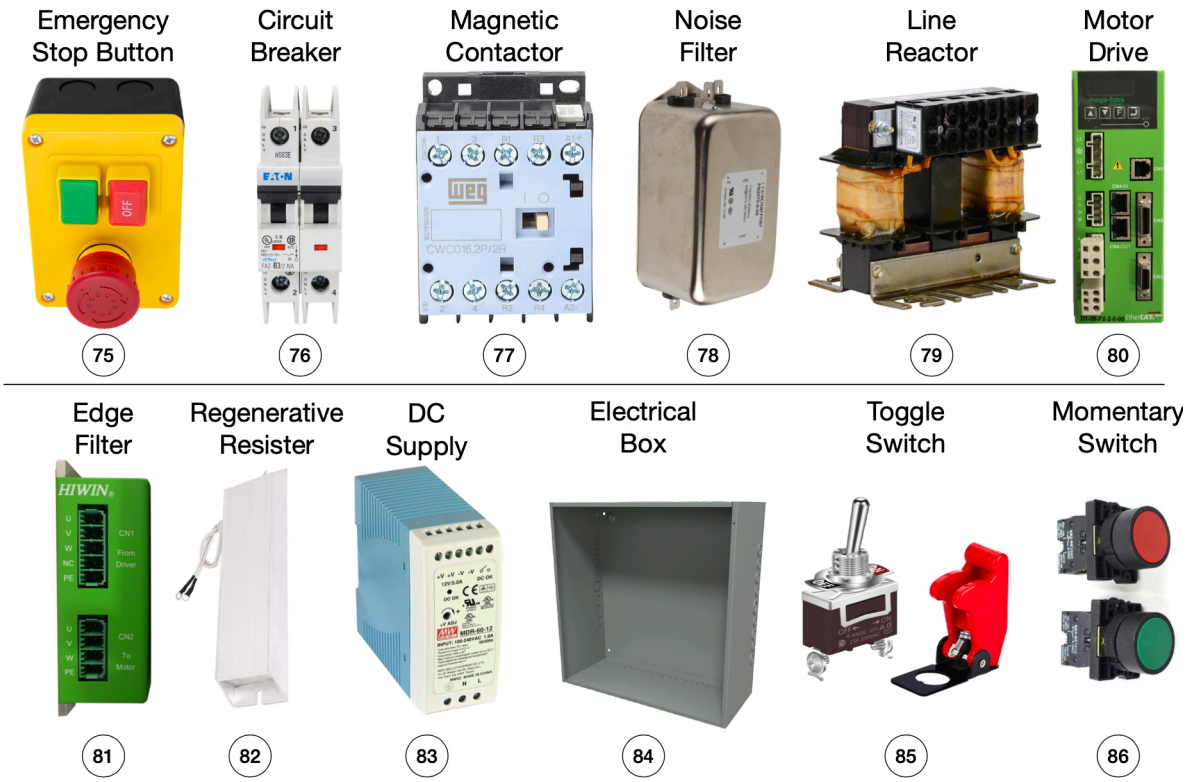


Figure 6.12: Electrical components needed to provide power supply to the linear motor. An electrical box is used to organize all the components.

brush block and differential driver. One end of the breadboard jumper cable is stripped away using a wire stripper and soldered onto the slip-ring brush block. The other end is directly inserted into the differential driver, as shown in Fig.6.11 (B). As for the slip-ring, it is first installed onto the pendulum shaft. The unterminated slip-ring wire is striped and then pushed into the pendulum arm using the groove and hole in the front of the pendulum arm, as shown in Fig.6.11 (C, D). Inside the pendulum arm, the slip-ring wire is clipped using a 3D printed cable management tool (5). Next, the other end is soldered with the locking clip contact pin, and the locking clip contact pin is insulated with a heat shrink tube to avoid the shortage of the wire connection. Finally, it is connected with the encoder to allow the transmission of the encoder signal through the slip-ring. Fig.6.11 (C, D, E) illustrates the slip-ring wiring. Details of the wiring diagram are in Appendix. C.5.1.

6.6.2 Electrical Part of Linear Motor Connection

The electrical part of the linear motor mainly concerns the following: 1) properly connecting the motor drive and motor; and 2) connecting the motor drive to the target computer. The connection guidance of the linear motor and linear motor drive is detailed in the user manual of Mega-Fabs D1 Drive[4]. The user manual also shows the pin out of the control signal cable, which enables the communication between the motor drive and target computer. The main challenge of the electrical part of the linear motor is to reduce the effect of electromagnetic interference (EMI) generated by the linear motor and motor drive. Several techniques can be used for this: 1) use a ground filter to filter out the noise generated by the linear motor and the drive in the ground line; 2) use shielded cables to transmit the signal; 3) use twisted pairs of wires to transmit differential signals; and 4) proper grounding of the all electronic components. With all these techniques combined, EMI is reduced and a proper and safe connection of the linear motor electrical parts is achieved.

Several electrical components are required to connect the linear motor. We use the components that are suggested in the user manual of the D1 motor drive: 1) emergency stop switch; 2) circuit breaker; 3) noise filter; 4) magnetic contactor; 5) line reactor; 6) motor drive; 7) edge filter; 8) regenerative resistor; 9) DC voltage supply; 10) linear motor; 11) electrical box and mounting plate; and 12) other miscellaneous parts that help connecting the electrical components, such as wires, connectors, cables, etc. A summary of the major electrical components can be seen in Fig. 6.12. The details of these parts are introduced in Appendix. C.5.2, along with a detailed wiring diagram.

6.7 Conclusions and Discussion

In this paper, we have introduced an experimental multi-link pendulum on a cart system. This system can be used to collect experimental data from the single, double, and

[4]Available on https://github.com/dynamicslab/MultiArm-Pendulum

triple pendulum. Moreover, the user can control the pendulum motion by actuating the cart via a linear motor. This makes the experimental system a powerful tool for studying the control of chaotic systems. Our experimental setup is open source, with all the detailed design choices and CAD files freely available, which allows reproduction of the system. We have also collected experimental data sets of the single, double, and triple pendulum and made them freely available. We believe this data will be valuable for the machine learning and modeling communities to test various algorithms. All code and design files can be downloaded on our Github page[5].

To make it possible to reproduce this pendulum on a cart system, we have provided a detailed description of the assembly and manufacturing process. Furthermore, the wiring diagrams of the electrical components and the software set up are also documented. Although the detailed manufacturing process of our setup is documented, we realize that many labs may not have the time and funding to replicate this system. Beyond this, the time needed to manufacture the experiment and the effort needed to maintain it may not be worth the investment for groups that only need occasional use of the experimental setup. This high cost can be mitigated by using more standard parts and 3D printing technology, but it does not solve the maintenance effort and the time consuming manufacturing and assembly process. This drawback motivates a future modification to our experimental set up to allow cloud access for remote users, discussed below, so that they can remotely access the system for data collection and control experiments. To conclude this manuscript we also briefly discuss the system operation procedures, safety instructions, and the estimated parameter values associated to the pendulum arms.

6.7.1 Software Setup

Two major software packages are used for successful and safe experiments: 1) The HIWIN Lightning software and 2) Simulink Real-Time model. The former is used to set

[5]Code available at https://github.com/dynamicslab/MultiArm-Pendulum

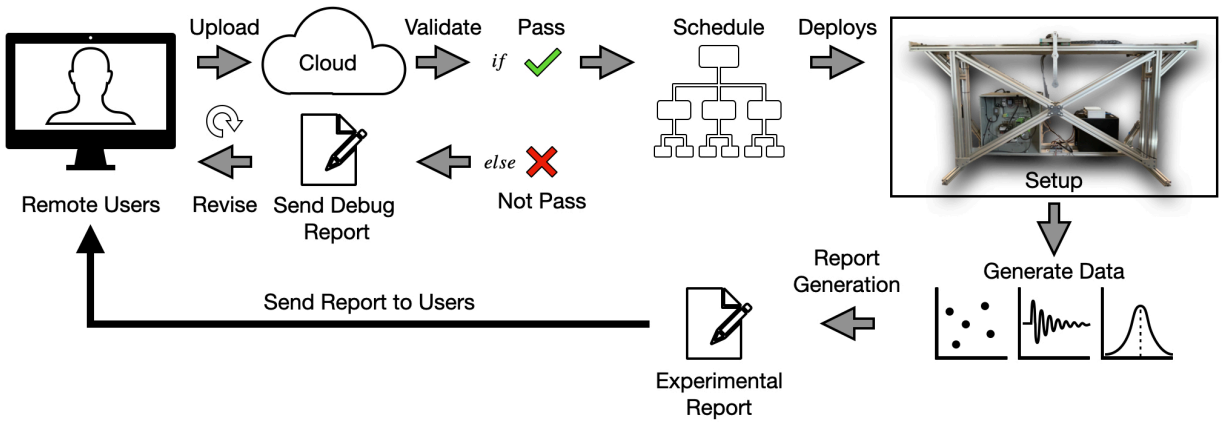


Figure 6.13: Overall schematic of a cloud experiment of the pendulum on the cart system.

up the parameters of the linear drive, while the latter is used to develop the real-time control algorithm for the system. The motor drive's Lightning software allows the selection of motor type and motor parameters, for which the motor drive uses this information to determine the control parameters needed to move the linear motor in the desired motion profile. The Lightnight software also configures the linear motor's encoder reading, allows the setup of the linear motor's Hall sensor, allows the user to select which in mode the linear motor should operate (position, velocity, force/torque, or stand-alone mode), configures the programmable I/Os of the D1 drive's CN2 channel, and finally, can set up software safety mechanisms. Appendix C.6.1 enumerates the details to setup these functionalities. The Simulink Real-Time software is used to develop the controller for the real-time pendulum experiments. To read the sensor signal and output control signal, the Simulink Real-Time and Speedgoat needs to be configured properly. The main aspects to be considered when setting up the Simulink model are: 1) setting up the Simulink blocks to allow the desired digital I/O functionalities, 2) setting up the FPGA module to enable encoder reading, 3) utilizing the digital I/Os to start the linear motor and stop it when the limit switches are triggered, 4) switching of the operating mode of the linear motor drive and activating homing function, 5) sending the analog signal to control the velocity of the linear motorm and 6) reading encoder sensors. These aspects are detailed in greater depth

in Appendix. C.6.2.

6.7.2 Operation and Safety

The operation procedures of the system include pre-experiment preparations, operations required during the experiments, and post-experiment operations. During the pre-experiment preparations, the operator should check all the wiring connections of the system and make sure there is no electrical shortage and hardware damage before starting. Then the controller file is prepared, and the system is turned on while ensuring safety. The main operation that must be performed during the experiment is to check whether the experiments are going as planned. If not, the operator should stop the experiments immediately and cut off all the power supply to the system. After the experiment, inspect any damage to the setup and then cut off all power to avoid any electrical hazards. For a detailed step-by-step operation guidance see Appendix. C.7. While operating the system, safety should always be the number one priority. To ensure the safe use of the experimental setup, mechanical, electrical, software, and personal safety measures must be implemented. A critical mechanical safety measure is to install a shock absorber on the side of the linear motor rail. This will help absorb the extra kinetic energy of the linear motor in case of controller failure. We further recommend purchasing a protective panel to surround the experimental setup. This can reduce the risk of personal injury caused by pendulum parts detaching while the linear motor is in motion. Appendix C.8 describes other safety measures that must be followed. Besides the safety notes in this chapter, the reader must also follow all the safety instructions written in the individual components' user manuals. We close this section by emphasizing that the proposed design of the multi-link pendulum on a cart has been tested and used safely in the lab environment by the authors. One should always exercise caution when building and operating the system and the authors can not be held responsible for any damage or injury caused by reproducing the design shown in this chapter.

6.7.3 Parameter Estimation of Pendulum Arms

Estimating the parameters of the pendulum system requires data collected from the designed system. An optimization problem is solved to find the pendulum system's parameters that best predict the data. This also requires an analytical model of the pendulum, which is shown in Appendix C.9.Parameter estimation is a standard task in pendulum control experiments where the model of the system is needed along with its parameters, while the model derivation of the single, double, and triple pendulum all follows through a similar process. We omit the model derivation for single and triple pendulums since they are similar. The former is standard and the latter can be derived through similar methods to the double pendulum [7, 428, 446].

In the case of the experimental pendulum, there are several parameters that must be identified, including the mass of the pendulum arm m and the position of center of mass, defined as a. We also require the length of the pendulum arm l and the inertial of the pendulum arm J, as depicted in Fig. 6.1. Although frequently ignored in parameter estimation, we also seek to determine the local constant of gravitational acceleration, g, which plays an important role on the chaotic dynamics of the double and triple pendulum. Some parameters do not show up in the derived equation of motion of the cart-pendulum, such as the mass of the cart M when the control is taken to be the acceleration of the cart. We summarize these definitions using the double pendulum as an example in Figure C.26 and Table. C.4 provides details of the estimated pendulum arm parameters. Finally, Table. C.4 also provides the values of the the friction coefficients ε_i which lead energy dissipation in the physical system. For more details on the method of parameter estimation, please see Appendix. C.9 for an exposition with the double pendulum.

6.7.4 Cloud Access to Experiments

Fig. 6.13 shows the remote cloud experiment concept. This is similar to cloud services where infrastructure, hardware, or software can be accessed by remote users through

the internet. In our case, the user accesses the pendulum hardware. The user must first develop their own program to achieve some objective using the experimental system. The program could be a controller to move the pendulum cart for data collection or for a control objective. This program should be coded according to a given template. After the user uploads their program to the cloud, it is necessary to test for any compiling errors and controller errors. Testing whether the proposed controller is safe is a difficult task. For example, some controllers might require the pendulum cart to move at unreasonable speeds or accelerations. If the controller is unstable, then it might damage the experimental setup. To avoid this, it may be necessary to test the controller in a digital twin of the pendulum. If position, velocity, or acceleration limits are violated, then a debug report will be sent to the user to allow revision and resubmission. Once the user program passes all tests on the digital twin, it may be deployed on the real experimental system. In the physical experiment, failsafe hardware limits may also be imposed, operating outside the user specified code.

During the experiment, the system should record all available data, including pendulum arm rotational angle and angular speed, linear motor position and velocity. The system should also record user defined variables. The collected data will be sent to the user to allow further analysis. A video camera will also record the experiments. The user will then be responsible for determining whether the desired objective is achieved by reviewing the recorded data and videos. The user should also be allowed to send a bug report to allow the maintenance of the hardware system.

There are several challenges that must be addressed for the cloud experiments. First is to deploy the user program to the real-time system automatically. In the current design, we use the Speedgoat machine as our real-time controller. Every time a controller is deployed, the user manually starts Simulink. A pipeline to automatically read, load, compile, and start the code must be developed. The second challenge is to automatically self-check the system. The real-time system should be able to perform the self-check and determine if any components must be replaced.

Chapter 7

SADDLE MEDIATED TRANSPORT OF DOUBLE PENDULUM

7.1 Introduction

The pendulum and heavenly bodies are two well-studied classical mechanics with centuries-old history. Their mechanics are governed by the simple physical law behind them, Newton's law of universal gravitation and Newton's second law for the celestial bodies, and laws of gravity and momentum for the pendulums. Albeit governed by the simple physical law, both dynamics can present rich, chaotic, and useful behaviors. For example, it has been shown that celestial bodies in the solar system can interact with each other and can be used to design fuel-efficient satellite and shuttle transport [439, 451–455] and provide evidence toward the existence of extreme trans-Neptunian objects [456–458]. Similarly, the pendulum, whose study can date back to Galileo, has long been used in timekeeping, estimating the constant of gravitational acceleration, or even studying chaos [371]. This similarity between the two classical mechanics is not accidental. In fact, when compared in detail, it becomes clear that the pendulum serves as an analog system to the celestial bodies.

The complexity of the multi-body problem increases as the number of celestial bodies increases. In its simplest form, the study of the two-body problem has a long tradition. The two-body problem constitutes the so-called Kepler problem and can successfully describe the elliptical motion of one body orbiting the other. The movement of a satellite orbiting a planet or a planet orbiting another planet can all be categorized into the two-body problem. When adding the third body into consideration, the dynamics of the resulting system become highly complicated and present chaos [459]. Poincaré first noticed this during his work on the three-body problem. Many simplifications are made, including

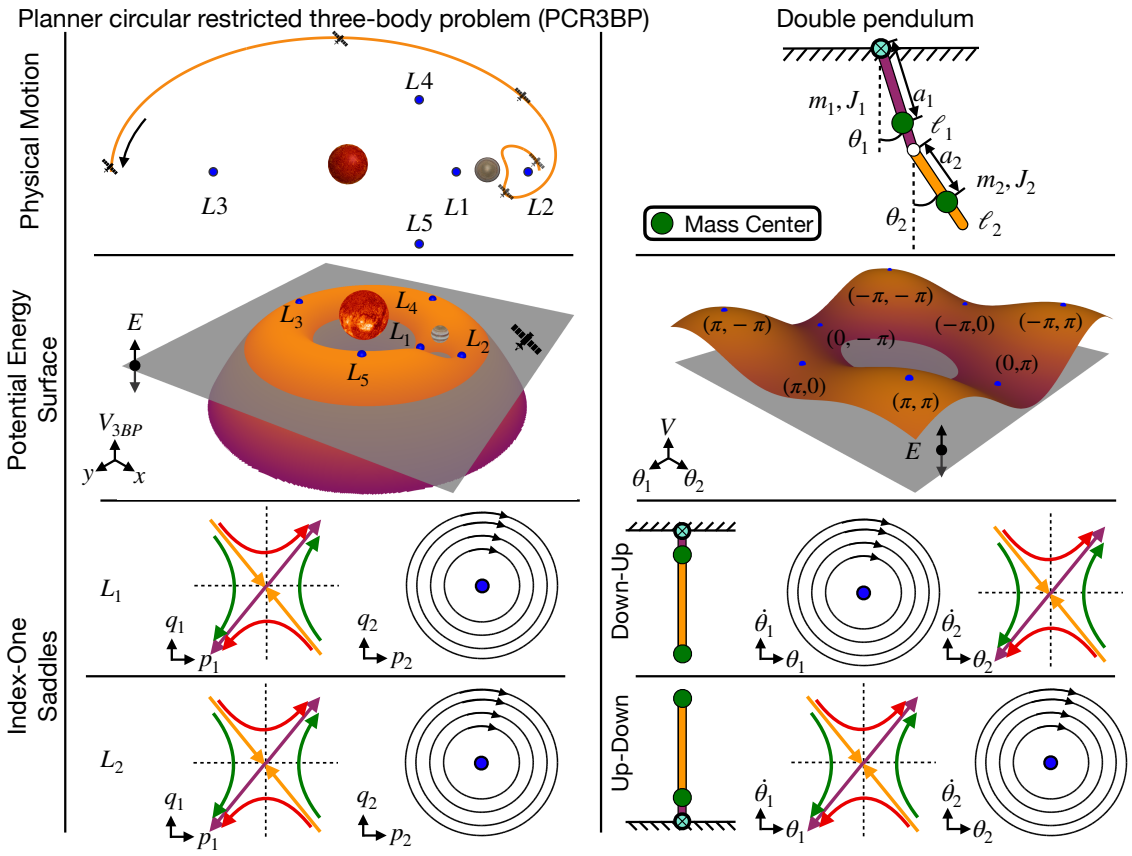


Figure 7.1: Despite representing dynamics on vastly different scales, the dynamics of the planar restricted 3-body problem (PCR3BP) and the double pendulum bear a striking resemblance. The Lagrange points L_1 and L_2 in the PCR3BP are analogous to the saddle Down-Up and Up-Down states since they all have one-dimensional stable and unstable directions and a two-dimensional center direction.

the planar circular restricted three-body problem (PCR3BP), to understand this complex behavior better. The PCR3BP consists of two bodies and a third body whose mass is significantly smaller than the other [437, 439, 451, 455]. As depicted in Figure 7.1, the PCR3BP has exactly five steady-state solutions, referred to as the Lagrange points and denoted L_1 through L_5. It is well-known that for a continuum of energies slightly above that of the Lagrange points L_1 and L_2 exist unstable periodic orbits (UPOs) [460, 461], known as Lyapunov orbits, that resemble halos about L_1 and L_2. These Lyapunov orbits

have two-dimensional stable and unstable manifolds, which we refer to throughout as *tubes* since they are diffeomorphic to a circle crossed with the real line. These Lyapunov orbits can be connected by homoclinic and heteroclinic orbits [437, 462–464], which allow one transits between the saddle points in phase space. The discovery of the tube structure in the PCR3BP has been a monumental step forward in our understanding of celestial dynamics and has prompted many to study more complex multi-body systems, again aimed at optimal space mission design [439, 451–454, 465]. Due to the abundance of these tubes in multi-body systems, they have been termed the *Interplanetary Transport Network* [466]. That is, these tubes provide gravitationally determined pathways that require little energy to follow and can be used to design itineraries that explore major bodies of our solar system.

The simplest form the pendulum is the single pendulum and it resembles the Kepler problem at low energies since all motion is periodic, representing the characteristic hypnotic swinging back and forth of a pendulum. As depicted in Figure 7.1, one can add another rod and body suspended from the body of the single pendulum to produce the *double pendulum*, whose study dates back at least to Bernoulli [381]. With just one pendulum arm added to the single pendulum, the double pendulum also starts to present chaos [384] with a much more complicated dynamics. The Kepler problem is to the single pendulum what the three-body problem is to the double pendulum. That is, like the three-body problem, the double pendulum has become a prototypical example of a chaotic system, and in both examples the simple base cases have little in common with their complex dynamics. Over the course of this Chapter we will demonstrate that this analogy is much more than superficial as the phase space dynamics of the double pendulum bears a striking resemblance to that of the PCR3BP. The double pendulum has two saddle steady-state solutions given by one arm hanging down and the other standing straight up, referred to as the Down-Up and Up-Down states, for which a neighborhood about these points is nearly identical to that of the Lagrange points L_1 and L_2 in the PCR3BP. We illustrate this correspondence in Figure 7.1 with a comparison of the potential energy landscapes of the

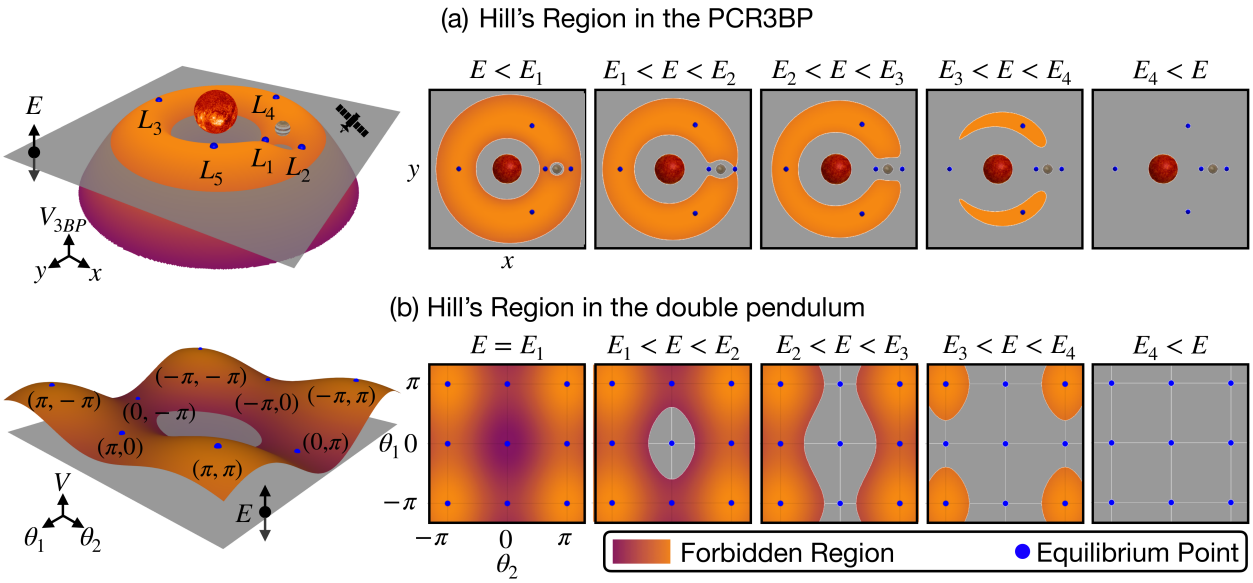


Figure 7.2: Hill's region comparison of PCR3BP and double pendulum.

PCR3BP and the double pendulum, a visualization of the Down-Up and Up-Down states, and an illustration of the dynamics near the saddle equilibria. We also demonstrate that Hill's region of the double pendulum shares similarities to the PCR3BP, as Fig. 7.2 shows. Using similar techniques to that of [437] for the PCR3BP, we demonstrate numerically the existence of homoclinic and heteroclinic orbits between the UPOs at energies slightly above those of the Down-Up and Up-Down steady-states. In particular, we demonstrate that the phase space of the double pendulum is similarly organized by global tube structures that can be exploited to enable macroscopic transport in the system with no additional energy expenditure.

Our work herein establishes the tube structure of the double pendulum. So it presents itself as a low-stakes and relatively low-cost testing ground to explore the saddle-mediated transport put forth for the PCR3BP in [437]. For a fraction of the cost of building a satellite or space shuttle, several researchers have built double pendulums [178, 384, 386, 390, 394–396, 417, 425, 444–446], including the pendulum hardware we built in Chapter 6, and so the theoretical work put forth here can and will be tested on these physical realizations of the

system in a follow-up investigation. Our work shows that following along a homoclinic orbit in the tube structures of the double pendulum represents the unstable Up arm of either the Down-Up or Up-Down saddles, making a full rotation before returning a neighborhood of the originating steady-state. Thus, stringing together an itinerary that follows homoclinic and heteroclinic connections would result in the physical double pendulum performing a sequence of acrobatic arm swinging motions while switching back and forth between the Up-Down and Down-Up states, resulting in the saddle mediated transport of double pendulum.

In this Chapter, we present a detailed study of the double pendulum while drawing comparisons to previous work on the PCR3BP. In particular, we follow a similar numerical procedure to [437, 439, 467] to determine the global tube structure emanating from the Down-Up and Up-Down saddle points of the double pendulum. We demonstrate that there exists homoclinic and heteroclinic connections that allow for macroscopic transport in both physical and state space.

7.2 Index-1 Saddles

Throughout this section we will seek to demonstrate the importance of saddle steady-states to the global dynamics of a Hamiltonian system. Throughout we denote the Hamiltonian function by $\mathcal{H}$ and only consider Hamiltonian systems with two degrees of freedom. The resulting dynamical system has a 4-dimensional phase space. Our interest will be in understanding the local dynamics near so-called *index-1* saddles, which are characterized by their linearization having two real eigenvalues, one positive and one negative, and a pair of complex conjugate purely imaginary eigenvalues. The terminology index-1 refers to the fact these orbits have one-dimensional stable and unstable manifolds associated to them.

7.2.1 *Linear Dynamics Near an Index-1 Saddle*

Let us begin by assuming that we have an equilibrium of the resulting 4-dimensional ODE and that linearizing about this equilibrium results in four eigenvalues $\pm\lambda$ and $\pm\mathrm{i}\omega$, for some real $\lambda, \omega > 0$. Through an invertible transformation we may translate this equilibrium to the origin and bring the resulting linearized dynamics into the Jordan normal form

$$\begin{bmatrix} \dot{p}_1 \\ \dot{q}_1 \\ \dot{p}_2 \\ \dot{q}_2 \end{bmatrix} = \begin{bmatrix} -\lambda & 0 & 0 & 0 \\ 0 & \lambda & 0 & 0 \\ 0 & 0 & 0 & -\omega \\ 0 & 0 & \omega & 0 \end{bmatrix} \cdot \begin{bmatrix} p_1 \\ q_1 \\ p_2 \\ q_2 \end{bmatrix}. \tag{7.1}$$

The linearized system (7.1) is itself a conservative system with quadratic Hamiltonian given by

$$\mathcal{H}_{\mathrm{loc}}(p_1, q_1, p_2, q_2) = \lambda p_1 q_1 + \frac{\omega}{2}(p_2^2 + q_2^2). \tag{7.2}$$

Solutions of (7.1) are easily determined to be

$$p_1(t) = p_1^0 \mathrm{e}^{-\lambda t}, \quad q_1(t) = q_1^0 \mathrm{e}^{\lambda t}, \quad p_2(t) + \mathrm{i}q_2(t) = (p_2^0 + \mathrm{i}q_2^0)\mathrm{e}^{-\mathrm{i}\omega t}, \tag{7.3}$$

for any constants $p_1^0, q_1^0, p_2^0, q_2^0 \in \mathbb{R}$. Importantly, (7.1) has three constants of motion

$$C_1 = p_1 q_1, \quad C_2 = p_2^2 + q_2^2, \quad C_3 = \mathcal{H}_{\mathrm{loc}} \tag{7.4}$$

which we will use to understand the flow near the saddle point.

From the above, we can see that projecting the dynamics into the (p_1, q_1)-plane results in a standard saddle in a planar system. Similarly, projecting into the (p_2, q_2)-plane results in a linear center consisting of harmonic oscillator motion. See Figure 7.1 for an illustration. For some fixed $h \in \mathbb{R}^+$ we can then consider the dynamics of (7.1) inside the level set

$\mathcal{H}_{\text{loc}} = h$. First, rearranging (7.2) gives

$$p_2^2 + q_2^2 = \frac{2}{\omega}(h - \lambda p_1 q_2), \tag{7.5}$$

so that in the bounding scenario that $C_1 = h/\lambda$ we have that the dynamics in the (p_2, q_2)-plane collapse to the point $(0, 0)$ and trajectories lie along the hyperbolas $p_1 q_1 = h/\lambda$. For $0 < C_1 < h/\lambda$ the dynamics in the (p_2, q_2)-plane lie along a circle and the trajectories are diffeomorphic to a circle crossed with a line.

When $p_1 = q_1 = 0$ the resulting dynamics are confined to the circles

$$S_h^1 := \left\{ (p_2, q_2) : \; p_2^2 + q_2^2 = \frac{2h}{\omega} \right\} \tag{7.6}$$

for each $h > 0$. These invariant circles constitute periodic orbits of the system, referred to as *Lyapunov orbits* in the context of celestial dynamics governed by many body equations. The above circles are only one of three nontrivial examples of trajectories with the constant of motion $C_1 = 0$. Simply restricting to the invariant manifold $q_1 = 0$ again gives $C_1 = 0$, but now the dynamics of p_1 need not be trivial. Using the solution (7.3) we find that $p_1(t) \to 0$ as $t \to \infty$ while the dynamics in the (p_2, q_2)-plane are still restricted to S_h^1. Therefore, the set $q_1 = 0$ is the stable manifold of the periodic orbit S_h^1, denoted $W^s(S_h^1)$. Similarly, taking $p_1 = 0$ and using the fact that $q_1(t) \to 0$ as $t \to -\infty$, we have that the set $p_1 = 0$ is the unstable manifold of the period orbit S_h^1, denoted $W^u(S_h^1)$. Both the stable and unstable manifolds of S_h^1 are homeomorphic to circles crossed with lines, which we henceforth refer to as tubes. Furthermore, the presence of both stable and unstable manifolds for the periodic orbits implies they are all saddles like the equilibrium they surround, and therefore are unstable.

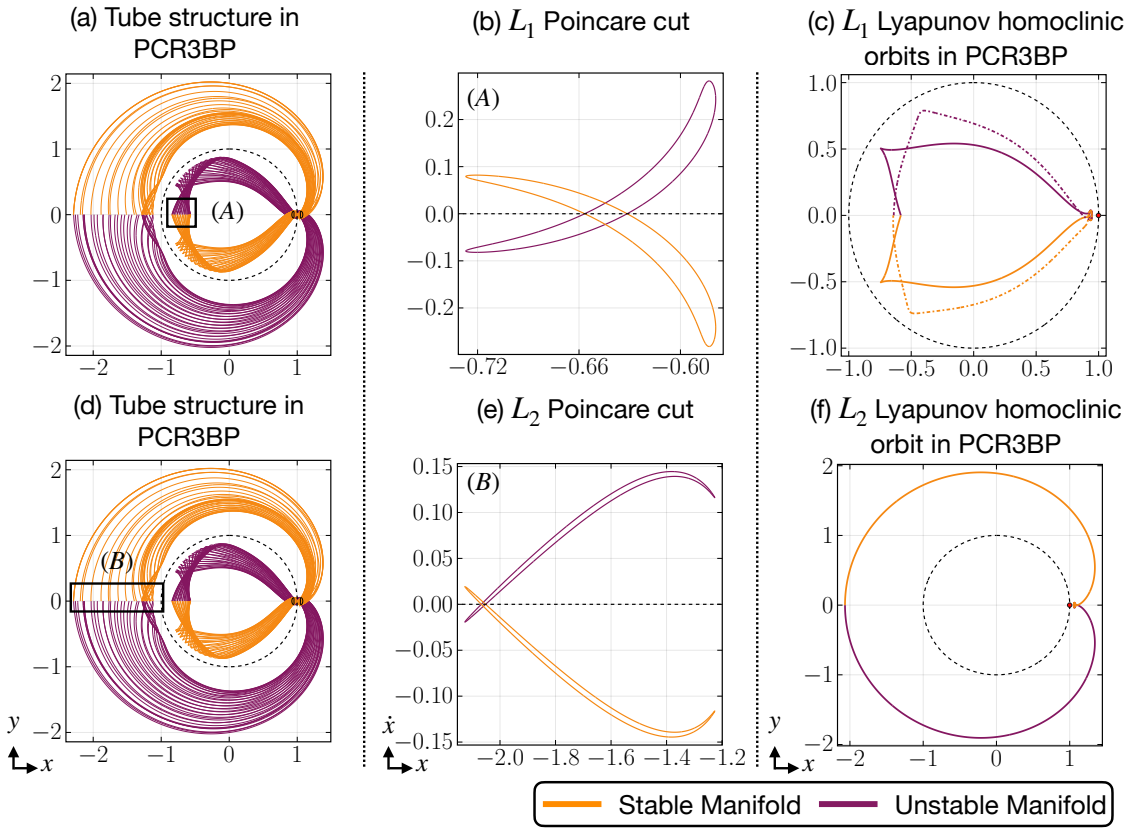


Figure 7.3: Numerically confirming the existence of homoclinic trajectories to the the L_1 and L_2 Lyapunov orbits in the PCR3BP is done by finding intersections between their stable and unstable manifolds in the Poincaré section $y = 0$ with $x < 0$. The manifolds are shown in (a) and (d), while their intersections with the Poincaré section are illustrated in (b) and (e). Intersections of the stable and unstable manifolds of each Lyapunov orbit results in a homoclinic orbit, some of which are depicted in (c) and (f). In (c), both symmetric (solid) and asymmetric (dashed) L_1 homoclinic orbits are shown.

7.2.2 Nonlinear Dynamics Near an Index-1 Saddle

In the previous subsection we described the linearized dynamics near an index-1 saddle point in a Hamiltonian system. We now extend this discussion to the nonlinear dynamics near the index-1 saddle. The same invertible change of variable that brings the linear dynamics to the system (7.1) transforms the Hamiltonian of the nonlinear system into the

form

$$\mathcal{H}(p_1, q_1, p_2, q_2) = \mathcal{H}_{\text{loc}}(p_1, q_1, p_2, q_2) + h.o.t., \tag{7.7}$$

where *h.o.t.* denotes the 'higher order terms' which in this case are at least cubic and $\mathcal{H}_{\text{loc}}$ is given in (7.2). Similarly, the dynamics in a neighborhood of the equilibrium (shifted to the origin in the p_1, q_1, p_2, q_2 variables) takes the form

$$\begin{bmatrix} \dot{p}_1 \\ \dot{q}_1 \\ \dot{p}_2 \\ \dot{q}_2 \end{bmatrix} = \begin{bmatrix} -\lambda & 0 & 0 & 0 \\ 0 & \lambda & 0 & 0 \\ 0 & 0 & 0 & -\omega \\ 0 & 0 & \omega & 0 \end{bmatrix} \cdot \begin{bmatrix} p_1 \\ q_1 \\ p_2 \\ q_2 \end{bmatrix} + h.o.t. \tag{7.8}$$

where now the higher order terms are at least quadratic. It is apparent from the fact that the linearization includes a complex conjugate pair of purely imaginary eigenvalues that one cannot apply results such as the Hartman–Grobman theorem to conclude that the dynamics (7.8) are locally conjugate to the dynamics of (7.1). Fortunately, the works [460, 461] exploit the Hamiltonian structure of the underlying dynamical system to provide the necessary results that demonstrate this. Precisely, there exists a small ball around $(p_1, q_1, p_2, q_2) = (0, 0, 0, 0)$ for which the dynamics of (7.8) are equivalent to the dynamics of (7.1).

The consequence of the above is that in a neighborhood of any index-1 saddle point is a continuum of UPOs. Furthermore, these UPOs have two-dimensional stable and unstable manifolds resembling tubes that eventually leave the region of validity for the conjugacy with the linear system (7.1). Using (7.7), it follows that these UPOs can only be guaranteed to exist for $\mathcal{H} = h$ with $0 < h \ll 1$ since the higher order terms become more important as h increases. By abuse of notation we will refer to these UPOs again as S^1_h since the UPOs of (7.1) and (7.8) lie in one-to-one correspondence when h is small. Moreover, the local components of the stable and unstable manifolds, $W^s_{\text{loc}}(S^1_h)$ and $W^u_{\text{loc}}(S^1_h)$, respectively, are approximated by the sets $\{q_1 = 0\}$ and $\{p_1 = 0\}$, respectively. These points become

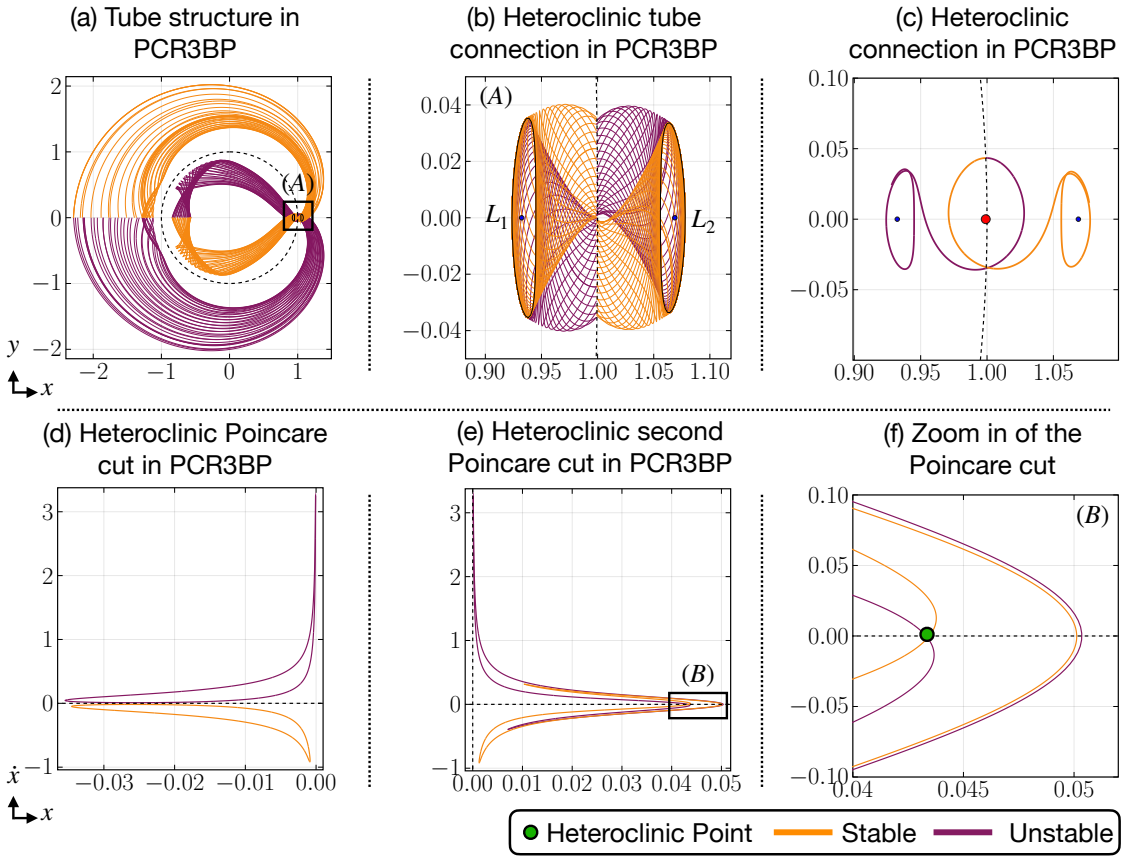


Figure 7.4: Heteroclinic orbits between the L_1 and L_2 Lyapunov orbits of the PCR3BP are found numerically in the same way as the homoclinic orbits in Figure 7.3, as demonstrated in (a) and (b). A heteroclinic orbit from an L_1 Lyapunov orbit to an L_2 Lyapunov orbit is presented in (c) and is identified from manifold intersections in the Poincaré section in (d). Similarly, (e) and (f) show manifold intersections in the Poincaré section resulting in heteroclinic orbits from an L_2 to an L_1 Lyapunov orbit.

important in the following sections for numerically following the global stable and unstable manifolds of different UPOs.

7.2.3 Global Tube Dynamics

The previous subsections focus on local aspects of the Hamiltonian phase space in neighborhoods of index-1 saddles. Per the previous discussion, the saddles and their

nearby UPOs all have stable and unstable manifolds that extend beyond these neighborhoods. Importantly, intersections of these stable and unstable manifolds give the existence of global structures such as homoclinic and heteroclinic orbits in the phase space. These homoclinic and heteroclinic orbits reveal the global geometry of phase space and to transport trajectories over vast regions of space with no extra energy expenditure.

Note that with a 2 degrees of freedom Hamiltonian system, we are restricted to the (generically) three-dimensional level sets of the Hamiltonian function. In one of these three-dimensional level sets the index-1 saddles have one-dimensional stable and unstable manifolds, meaning that the existence of a homoclinic orbit is unlikely since these stable and unstable manifolds would have to coincide. Such a phenomenon is not robust with respect to generic parameter manipulation, and so this situation should not be expected without additional assumptions such as symmetries put on the system. Alternatively, for a connected continuum of values of the Hamiltonian slightly above the value at the index-1 saddle we have the UPOs, each of which have two-dimensional stable and unstable manifolds inside the three-dimensional level sets. In this case an intersection of these manifolds is robust and expected to be one-dimensional, leading to the existence of a homoclinic orbit. The robustness of these intersections implies that the existence of a homoclinic orbit for one value of the Hamiltonian is expected to imply the existence for values of the Hamiltonian in an open neighborhood of said value. Heteroclinic connections between index-1 saddles are similarly robust and are made up of intersections of one UPO's stable manifold intersecting along a generically one-dimensional curve with another UPO's unstable manifold. We again expect the existence of such a heteroclinic connection to exist for a range of values of the Hamiltonian, if it exists at all.

The existence of homoclinic orbits to the UPOs near index-1 saddles has notably been proven for the Lyapunov orbits around the L_1 Lagrange point in the PCR3BP [463, 464]. These results were improved by [468] to show that under appropriate conditions one could conclude that these intersections were transverse, meaning that they are indeed robust. More in line with our work herein, the work [437] followed the ideas put forth by [467]

to explore the system numerically and detect the existence of these homoclinic orbits to not only Lyapunov orbits of the L_1 Lagrange point, but also the L_2 point. In a process that will be described in more detail below, one simulates the unstable manifold of the Lyapunov orbit forward in time and the stable manifold backward in time until they meet at an appropriately defined Poinacaré section. The intersection of these manifolds with the Poincaré section will be diffeomorphic to a circle and the intersection of these circles represent the desired homoclinic trajectories. This process and the found homoclinic orbits are illustrated in Figure 7.3. We refer to the Appendix. D.1 for details on the PCR3BP including the equations of motion, the location of the Lagrange points L_1 and L_2, and the parameter values used in our computations.

Beyond just the homoclinic orbits, [437] also numerically demonstrates the existence of heteroclinic orbits between the L_1 and L_2 Lyapunov orbits. These are obtained in the same way as the homoclinic orbits in that stable and unstable manifolds are simulated backward and forward in time, respectively, to meet an appropriate Poincaré section to find intersections of these manifolds. This process and the results are summarized in Figure 7.4. The effect this has is that for values of the Hamiltonian slightly above that of the index-1 saddles there exist level sets containing Lyapunov orbits to both L_1 and L_2 which each have homoclinic orbits and are connected by a heteroclinic orbit. These orbits are used in [437] to construct itineraries through phase space that allow one to jump from following the homoclinics to the heteroclinics and back, thus providing trajectories that allow the lightweight object to fully explore the phase space with no additional energy expenditure. In what follows we will perform a similar investigation for the double pendulum by identifying homoclinic and heteroclinic trajectories to index-1 saddles.

7.3 Equations of Motion

In this section we provide the relevant equations of motion for the physical double pendulum. We note that many mathematical investigations of the double pendulum use the *point-mass double pendulum*, which represents a point-mass model of the double pendulum

where the mass of the pendulum rod is ignored. To make our investigation applicable to future laboratory experiments, we will consider the *experimental double pendulum* which includes the mass of the pendulum arms. In Figure C.26 we provide a comparison between the theoretical and experimental double pendulum models and we direct the reader to the Appendix. D.2 for the equations of motion for the theoretical double pendulum for comparison with the equations presented in this section.

To begin, let us denote m_1 as the mass of the first pendulum arm and m_2 as the mass of second pendulum arm. Then, denote ℓ_1 and ℓ_2 as the length of the first and second arm, respectively. Next, define a_1 and a_2 as the positions of the centers of mass of the first and second pendulum arm, respectively. Let J_1 and J_2 denote the inertial of the first and second pendulum arm and designate g to be the constant of gravitational acceleration. Finally, we take θ_1 and θ_2 to be the time-dependent rotational angles of pendulum arms, as illustrated in Figure C.26.

With the above defined constants and variables, the position of centers of mass can be written in Cartesian coordinates as

$$\begin{aligned} x_1 &= a_1 \sin(\theta_1), \\ y_1 &= -a_1 \cos(\theta_1), \\ x_2 &= \ell_1 \sin(\theta_1) + a_2 \sin(\theta_2), \\ y_2 &= -\ell_1 \cos(\theta_1) - a_2 \cos(\theta_2). \end{aligned} \tag{7.9}$$

Using (7.9), the velocity of the mass centers can be found by differentiating with respect to time, giving

$$\begin{aligned} v_{x_1} &= a_1 \cos(\theta_1)\dot{\theta}_1, \\ v_{y_1} &= a_1 \sin(\theta_1)\dot{\theta}_1, \\ v_{x_2} &= \ell_1 \cos(\theta_1)\dot{\theta}_1 + a_2 \cos(\theta_2)\dot{\theta}_2, \\ v_{y_2} &= \ell_1 \sin(\theta_1)\dot{\theta}_1 + a_2 \sin(\theta_2)\dot{\theta}_2. \end{aligned} \tag{7.10}$$

With the positions (7.9) and velocities (7.10), we then have that the kinetic energy of the experimental double pendulum is given by

$$T = \frac{1}{2}(m_1(v_{x_1}^2 + v_{y_1}^2) + m_2(v_{x_2}^2 + v_{y_2}^2) + J_1\dot{\theta}_1^2 + J_2\dot{\theta}_2^2). \tag{7.11}$$

Similarly, the potential energy of the double pendulum is found to be

$$\begin{aligned} V &= (m_1 y_1 + m_2 y_2)g \\ &= (-m_1 a_1 \cos(\theta_1) - m_2(\ell_1 \cos(\theta_1) + a_2 \cos(\theta_2)))g. \end{aligned} \tag{7.12}$$

Using (7.11) and (7.12), the Lagrangian of the experimental double pendulum can be calculated using

$$L = T - V, \tag{7.13}$$

and the Hamiltonian of the system can be given by

$$\mathcal{H} = T + V. \tag{7.14}$$

Throughout our investigation in this manuscript we will neglect the effect of friction.

Using the Lagrangian (7.13), the equations of motion of the double pendulum can be obtained by the formulas

$$\frac{\partial}{\partial t}\frac{\partial L}{\partial \dot{\theta}_1} - \frac{\partial L}{\partial \theta_1} = 0, \tag{7.15a}$$

$$\frac{\partial}{\partial t}\frac{\partial L}{\partial \dot{\theta}_2} - \frac{\partial L}{\partial \theta_2} = 0. \tag{7.15b}$$

Equation (7.15a) yields

$$\begin{aligned} 0 = J_1\ddot{\theta}_1 + \ell_1^2\ddot{\theta}_1 m_2 + a_1^2\ddot{\theta}_1 m_1 + \ell_1 g m_2 \sin(\theta_1) + a_1 g m_1 \sin(\theta_1) \\ + \ell_1 a_2 \dot{\theta}_2^2 m_2 \sin(\theta_1 - \theta_2) + \ell_1 a_2 \ddot{\theta}_2 m_2 \cos(\theta_1 - \theta_2), \end{aligned} \tag{7.16}$$

while equation (7.15b) yields

$$0 = J_2\ddot{\theta}_2 - \ell_1 a_2 \dot{\theta}_1^2 m_2 \sin(\theta_1 - \theta_2) + \ell_1 a_2 \ddot{\theta}_1 m_2 \cos(\theta_1 - \theta_2), \tag{7.17}$$

From (7.16) and (7.17) we can solve for $\ddot{\theta}_1$ and $\ddot{\theta}_2$, giving

$$\begin{aligned} \ddot{\theta}_1 &= \frac{A_{10}\sin(\theta_1) + A_{11}\sin(\theta_1 - \theta_2) + A_{12}\sin(\theta_1 - 2\theta_2) + A_{22}\sin(2\theta_1 - 2\theta_2)}{D(\theta_1 - \theta_2)} \\ \ddot{\theta}_2 &= \frac{B_{01}\sin(\theta_2) + B_{11}\sin(\theta_1 - \theta_2) + B_{21}\sin(2\theta_1 - \theta_2) + B_{22}\sin(2\theta_1 - 2\theta_2)}{D(\theta_1 - \theta_2)} \end{aligned} \tag{7.18}$$

where

$$\begin{aligned} A_{10} &= -\ell_1 g a_2^2 m_2^2 - 2a_1 g m_1 a_2^2 m_2 - 2J_2 l_1 g m_2 - 2J_2 a_1 g m_1 \\ A_{11} &= -2\ell_1 a_2^3 \dot{\theta}_2^2 m_2^2 - 2J_2 \ell_1 a_2 \dot{\theta}_2^2 m_2 \\ A_{12} &= -\ell_1 a_2^2 g m_2^2 \\ A_{22} &= -\ell_1^2 a_2^2 \dot{\theta}_1^2 m_2^2 \\ B_{01} &= -a_2 m_2 (g m_2 \ell_1^2 - g m_1 \ell_1 a_1 + 2g m_1 a_1^2 + 2J_1 g) \\ B_{11} &= a_2 m_2 (2m_2 \ell_1^3 \dot{\theta}_1^2 + 2m_1 \ell_1 a_1^2 \dot{\theta}_1^2 + 2J_1 \ell_1 \dot{\theta}_1^2) \\ B_{21} &= a_2 m_2 (g m_2 \ell_1^2 + a_1 g m_1 \ell_1) \\ B_{22} &= 2m_2 (g m_2 \ell_1^2 + a_1 g m_1 \ell_1) \\ D(\theta_1 - \theta_2) &= -\ell_1^2 a_2^2 m_2^2 cos(\theta_1 - \theta_2)^2 + \ell_1^2 a_2^2 m_2^2 + J_2 \ell_1^2 m_2 + m_1 a_1^2 a_2^2 m_2 \\ &\quad + J_2 m_1 a_1^2 + J_1 a_2^2 m_2 + J_1 J_2 \end{aligned} \quad . \tag{7.19}$$

Here we use the convention that A_{ij}, B_{ij} are the coefficients of the energy preserved Hamiltonian system for the $\ddot{\theta}_1$ and $\ddot{\theta}_2$ equations, respectively, while the subscripts denote the coefficients on the θ_1 and θ_2 terms inside the sine function they are multiplied against. The notation $D(\theta_1 - \theta_2)$ represents the function in the denominator of both equations.

Writing (7.18) as a first order ODE gives

$$\begin{aligned}
\dot{\theta}_1 &= \omega_1, \\
\dot{\theta}_2 &= \omega_2, \\
\dot{\omega}_1 &= \frac{A_{10}\sin(\theta_1) + A_{11}\sin(\theta_1-\theta_2) + A_{12}\sin(\theta_1-2\theta_2) + A_{22}\sin(2\theta_1-2\theta_2)}{D(\theta_1-\theta_2)}, \\
\dot{\omega}_2 &= \frac{B_{01}\sin(\theta_2) + B_{11}\sin(\theta_1-\theta_2) + B_{21}\sin(2\theta_1-\theta_2) + B_{22}\sin(2\theta_1-2\theta_2)}{D(\theta_1-\theta_2)},
\end{aligned} \tag{7.20}$$

System (7.20) therefore represents the full equations of motion for the double pendulum that we investigate for the remainder of this manuscript.

Throughout this manuscript our numerical experiments will use the parameter values (with relevant units)

$$\begin{aligned}
&m_1 = 0.0938\text{kg}, \quad m_2 = 0.1376\text{kg}, \quad a_1 = 0.1086\text{m}, \quad a_2 = 0.1168\text{m}, \\
&\ell_1 = 0.1727\text{m}, \qquad J_1 = 10^{-4}\text{kgm}^2, \quad J_2 = 10^{-4}\text{kgm}^2, \quad g = 9.808\text{m/s}^2.
\end{aligned} \tag{7.21}$$

All of the above values were obtained via parameter estimation from the physical model constructed in Chapter. 6 and chosen to demonstrate that the results of this manuscript are physically realizable. For more details on the parameter estimation technique we used to ontain those parameters, please check Appendix. C.9. We have also found that other similar choices of parameter values lead to nearly identical results and so we do not expect that the restriction to these parameter values is a limitation of this work. Finally, notice that the length of the second arm, ℓ_2, is notably absent from the equations of motion (7.20) and so its value is not necessary for our investigation of the experimental double pendulum.

7.4 Tube Dynamics of the Double Pendulum

In this section we will consider the tube structure of transport between index-1 saddles of system (7.20). Over the following subsections we seek to identify its index-1 saddles and their global tube dynamics. This will help us to identify the macroscopic transport

mechanisms of the the experimental double pendulum modelled by equations (7.20) in a similar manner to what has been done for a variety of other Hamiltonian systems.

We begin in §7.4.1 by describing the reversible symmetry of (7.20). This reversible symmetry will help us to classify the trajectories orbits between neighborhoods of index-1 saddles as either symmetric or asymmetric, with those that are asymmetric coming in pairs related by applying the symmetry transformation. We then proceed to identify the index-1 saddles of (7.20) in §7.4.2 by analyzing the linearization about the system about each of its steady-states. We identify two index-1 saddles given by one of the pendulum arms standing straight up and the other hanging down. We provide a numerical justification for the existence of homoclinic orbits to UPOs in the neighborhoods of these saddles in §7.4.3 and §7.4.4. We then conclude this section in §7.4.5 by numerically demonstrating the existence of heteroclinic orbits between UPOs in the neighborhoods of each of the index-1 saddles.

7.4.1 Reversible Symmetry

Beyond the Hamiltonian structure of (7.20), we note that the system is also reversible in the sense that if $(\theta_1(t), \theta_2(t), \omega_1(t), \omega_2(t))$ is a solution, then so is $(\theta_1(-t), \theta_2(-t), -\omega_1(-t), -\omega_2(-t))$. Precisely, we define the reverser

$$\mathcal{R} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix} \tag{7.22}$$

and say that $\Theta(t) := (\theta_1(t), \theta_2(t), \omega_1(t), \omega_2(t))$ and $\mathcal{R}\Theta(-t)$ are solutions. Using this property, we will refer to a solution as *symmetric* if $\Theta(t) = \mathcal{R}\Theta(-t)$ for all $t \in \mathbb{R}$. Importantly, if Θ_* is a symmetric equilibrium solution of (7.20), i.e. $\mathcal{R}\Theta_* = \Theta_*$, then the associated stable manifold $W^s(\Theta_*)$ and unstable manifold $W^u(\Theta_*)$ are such that $W^s(\Theta_*) = \mathcal{R}W^u(\Theta_*)$.

Therefore, homoclinic orbits of symmetric equilibria are either symmetric or asymmetric, for which the latter case implies that they come in pairs, related by applying $\mathcal{R}$ and reversing the flow of time. The preceding discussion also holds for the symmetric UPOs near a symmetric index-1 saddle which are the focus of much of our discussion going forward. We will therefore emphasize the symmetric and asymmetric homoclinic orbits that can be observed in our numerical findings.

7.4.2 Linear Analysis

From the equations of motion (7.20) we can see that equilibria come in the form $(\theta_1, \theta_2, \omega_1, \omega_2) = (\pi k_1, \pi k_2, 0, 0)$ for every pair of integers $(k_1, k_2) \in \mathbb{Z}^2$. All of these equilibria are symmetric with respect to the action of the reverser $\mathcal{R}$, defined in (7.22). Although we have obtained a lattice of equilibria, periodicity of the components $\theta_{1,2}$ implies that we may restrict our analysis to the four distinct steady-states

$$\begin{aligned} \text{Down} - \text{Down}: &\quad (\theta_1, \theta_2, \omega_1, \omega_2) = (0, 0, 0, 0) \\ \text{Down} - \text{Up}: &\quad (\theta_1, \theta_2, \omega_1, \omega_2) = (0, \pi, 0, 0) \\ \text{Up} - \text{Down}: &\quad (\theta_1, \theta_2, \omega_1, \omega_2) = (\pi, 0, 0, 0) \\ \text{Up} - \text{Up}: &\quad (\theta_1, \theta_2, \omega_1, \omega_2) = (\pi, \pi, 0, 0) \end{aligned} . \tag{7.23}$$

The naming of the states represents the vertical position of the arms of the double pendulum. Precisely, Down-Down has both arms hanging straight down, Down-Up has the first arm with parameters (ℓ_1, m_1) hanging down and the other standing straight up, Up-Down has the first arm standing straight up and the other hanging down, and the Up-Up state has both arms standing straight up. We note that we have listed the steady-states in (7.23) in order of ascending energy values, according to the Hamiltonian (7.14). Our results that follow apply for all choices of parameters, not just those provided in (7.21).

Linearizing (7.20) about an equilibrium $(\theta_1, \theta_2, \omega_1, \omega_2) = (\pi k_1, \pi k_2, 0, 0)$ with $k_1, k_2 \in \mathbb{Z}$

results in the matrix of the form

$$\begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \frac{\sigma_2\sigma_3}{\sigma_1^2} - \frac{\sigma_4}{2\sigma_1} & \frac{\sigma_5}{\sigma_1} - \frac{\sigma_2\sigma_3}{\sigma_1^2} & 0 & 0 \\ \frac{\sigma_6}{\sigma_1} - \frac{\sigma_7\sigma_8}{\sigma_1^2} & -\frac{\sigma_9}{\sigma_1} - \frac{\sigma_7\sigma_8}{\sigma_1^2} & 0 & 0 \end{bmatrix}. \tag{7.24}$$

with

$$\begin{aligned}
\sigma_1 &= -\ell_1^2 a_2^2 m_2^2 \cos(k_1\pi - k_2\pi)^2 + \ell_1^2 a_2^2 m_2^2 \\
&\quad + J_2 \ell_1^2 m_2 + m_1 a_1^2 a_2^2 m_2 + J_2 m_1 a_1^2 + J_1 a_2^2 m_2 + J_1 J_2, \\
\sigma_2 &= \ell_1^2 a_2^2 g m_2^2 \cos(k_1\pi - k_2\pi)\sin(k_1\pi - k_2\pi), \\
\sigma_3 &= (\ell_1 a_2^2 m_2^2 \sin(k_1\pi) + 2J_2 \ell_1 m_2 \sin(k_1\pi) \\
&\quad + \ell_1 a_2^2 m_2^2 \sin(k_1\pi - 2k_2\pi) + 2J_2 a_1 m_1 \sin(k_1\pi) + 2a_1 a_2^2 m_1 m_2 \sin(k_1\pi)), \\
\sigma_4 &= (\ell_1 a_2^2 m_2^2 \cos(k_1\pi) + 2J_2 \ell_1 m_2 \cos(k_1\pi) \\
&\quad + \ell_1 a_2^2 m_2^2 \cos(k_1\pi - 2k_2\pi) + 2J_2 a_1 m_1 \cos(k_1\pi) + 2a_1 a_2^2 m_1 m_2 \cos(k_1\pi)), \\
\sigma_5 &= \ell_1 a_2^2 g m_2^2 \cos(k_1\pi - 2k_2\pi), \\
\sigma_6 &= \ell_1 a_2 g m_2 \cos(2k_1\pi - k_2\pi)(\ell_1 m_2 + a_1 m_1), \\
\sigma_7 &= 2\ell_1^2 a_2^3 g m_2^3 \cos(k_1\pi - k_2\pi)\sin(k_1\pi - k_2\pi), \\
\sigma_8 &= (J_1 \sin(k_2\pi) + \ell_1^2 m_2 \sin(k_2\pi) + a_1^2 m_1 \sin(k_2\pi) \\
&\quad - \ell_1^2 m_2 \cos(k_1\pi - k_2\pi)\sin(k_1\pi) - \ell_1 a_1 m_1 \cos(k_1\pi - k_2\pi)\sin(k_1\pi)), \\
\sigma_9 &= a_2 g m_2 (J_1 \cos(k_2\pi) + \ell_1^2 m_2 \cos(k_2\pi) + a_1^2 m_1 \cos(k_2\pi) \\
&\quad - \ell_1^2 m_2 \sin(k_1\pi - k_2\pi)\sin(k_1\pi) - \ell_1 a_1 m_1 \sin(k_1\pi - k_2\pi)\sin(k_1\pi)).
\end{aligned} \tag{7.25}$$

Due to the block structure of (7.24), the square of its eigenvalues are equal to those of the lower left 2×2 matrix

$$\begin{bmatrix} \frac{\sigma_2\sigma_3}{\sigma_1^2} - \frac{\sigma_4}{2\sigma_1} & \frac{\sigma_5}{\sigma_1} - \frac{\sigma_2\sigma_3}{\sigma_1^2} \\ \frac{\sigma_6}{\sigma_1} - \frac{\sigma_7\sigma_8}{\sigma_1^2} & -\frac{\sigma_9}{\sigma_1} - \frac{\sigma_7\sigma_8}{\sigma_1^2} \end{bmatrix}. \tag{7.26}$$

We will now proceed to classify the stability of the four equilibria in (7.23) using the matrix (7.26).

Down-Down. The Down-Down state is obtained by setting $k_1 = k_2 = 0$. In this case (7.26) becomes

$$\begin{bmatrix} -\frac{g(m_2a_2^2+J_2)(\ell_1m_2+a_1m_1)}{J_2m_2\ell_1^2+m_1m_2a_1^2a_2^2+J_2m_1a_1^2+J_1m_2a_2^2+J_1J_2} & \frac{\ell_1a_2^2gm_2^2}{J_2m_2\ell_1^2+m_1m_2a_1^2a_2^2+J_2m_1a_1^2+J_1m_2a_2^2+J_1J_2} \\ \frac{\ell_1a_2gm_2(\ell_1m_2+a_1m_1)}{J_2m_2\ell_1^2+m_1m_2a_1^2a_2^2+J_2m_1a_1^2+J_1m_2a_2^2+J_1J_2} & -\frac{a_2gm_2(m_2\ell_1^2+m_1a_1^2+J_1)}{J_2m_2\ell_1^2+m_1m_2a_1^2a_2^2+J_2m_1a_1^2+J_1m_2a_2^2+J_1J_2} \end{bmatrix}. \tag{7.27}$$

Since the trace of the above matrix is negative and the determinant is positive for all the physical parameters m_1,m_2,a_1,a_2,J_1,J_2,g,ℓ_1,ℓ_2, it follows that both eigenvalues are distinct and strictly negative. Therefore, the linearization (7.24) about the Down-Down state has two pairs of purely complex eigenvalues, making it a linear center.

Down-Up. The Down-Up state is obtained by setting $k_1 = 0$ and $k_2 = 1$. The matrix (7.26) becomes

$$\begin{bmatrix} -\frac{g(m_2a_2^2+J_2)(\ell_1m_2+a_1m_1)}{J_2m_2\ell_1^2+m_1m_2a_1^2a_2^2+J_2m_1a_1^2+J_1m_2a_2^2+J_1J_2} & \frac{\ell_1a_2^2gm_2^2}{J_2m_2\ell_1^2+m_1m_2a_1^2a_2^2+J_2m_1a_1^2+J_1m_2a_2^2+J_1J_2} \\ -\frac{\ell_1a_2gm_2(\ell_1m_2+a_1m_1)}{J_2m_2\ell_1^2+m_1m_2a_1^2a_2^2+J_2m_1a_1^2+J_1m_2a_2^2+J_1J_2} & \frac{a_2gm_2(m_2\ell_1^2+m_1a_1^2+J_1)}{J_2m_2\ell_1^2+m_1m_2a_1^2a_2^2+J_2m_1a_1^2+J_1m_2a_2^2+J_1J_2} \end{bmatrix}. \tag{7.28}$$

which has a negative determinant for all relevant parameter values. Therefore, the above matrix has one positive and one negative real eigenvalue along with a pair of purely imaginary eigenvalues. Hence, the Down-Up state is an index-1 saddle.

Up-Down. We obtain the Up-Down state by setting $k_1 = 1$ and $k_2 = 0$. The matrix (7.26) is then given by

$$\begin{bmatrix} \frac{g(m_2a_2^2+J_2)(\ell_1m_2+a_1m_1)}{J_2m_2\ell_1^2+m_1m_2a_1^2a_2^2+J_2m_1a_1^2+J_1m_2a_2^2+J_1J_2} & -\frac{\ell_1a_2^2gm_2^2}{J_2m_2\ell_1^2+m_1m_2a_1^2a_2^2+J_2m_1a_1^2+J_1m_2a_2^2+J_1J_2} \\ \frac{\ell_1a_2gm_2(\ell_1m_2+a_1m_1)}{J_2m_2\ell_1^2+m_1m_2a_1^2a_2^2+J_2m_1a_1^2+J_1m_2a_2^2+J_1J_2} & -\frac{a_2gm_2(m_2\ell_1^2+m_1a_1^2+J_1)}{J_2m_2\ell_1^2+m_1m_2a_1^2a_2^2+J_2m_1a_1^2+J_1m_2a_2^2+J_1J_2} \end{bmatrix}. \tag{7.29}$$

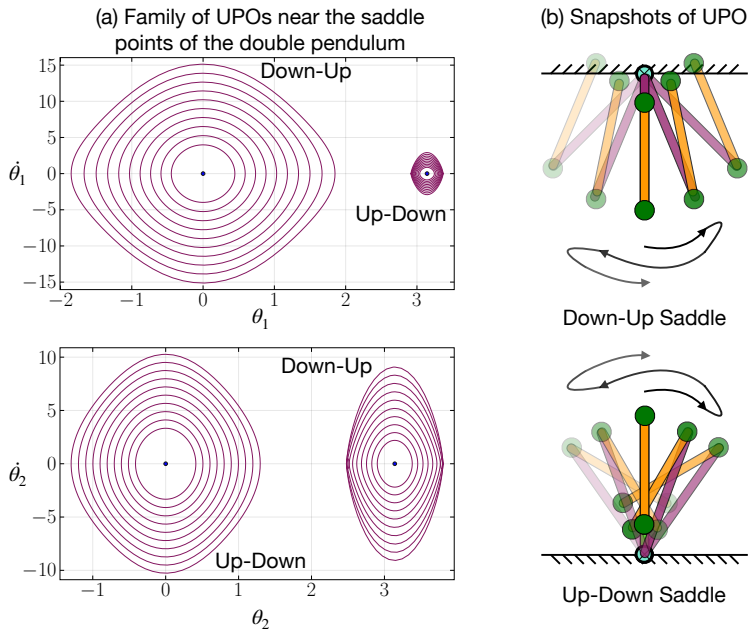


Figure 7.5: Near the index-1 saddles of the double pendulum there are infinitely many UPOs. These UPOs are projected into (a) the $(\theta_1, \dot{\theta}_1)$-plane and the $(\theta_2, \dot{\theta}_2)$-plane with a cartoon of their physical motion in the double pendulum shown in (b).

and following as in the Down-Up state, we find that the Up-Down state is also an index-1 saddle.

Up-Up. The Up-Up state has $k_1 = k_2 = 1$, resulting in (7.26) taking the form

$$\begin{bmatrix} \frac{g(m_2a_2^2+J_2)(\ell_1m_2+a_1m_1)}{J_2m_2\ell_1^2+m_1m_2a_1^2a_2^2+J_2m_1a_1^2+J_1m_2a_2^2+J_1J_2} & -\frac{\ell_1a_2^2gm_2^2}{J_2m_2\ell_1^2+m_1m_2a_1^2a_2^2+J_2m_1a_1^2+J_1m_2a_2^2+J_1J_2} \\ -\frac{\ell_1a_2gm_2(\ell_1m_2+a_1m_1)}{J_2m_2\ell_1^2+m_1m_2a_1^2a_2^2+J_2m_1a_1^2+J_1m_2a_2^2+J_1J_2} & \frac{a_2gm_2(m_2\ell_1^2+m_1a_1^2+J_1)}{J_2m_2\ell_1^2+m_1m_2a_1^2a_2^2+J_2m_1a_1^2+J_1m_2a_2^2+J_1J_2} \end{bmatrix}. \tag{7.30}$$

The above matrix is the result of negating all of the entries of the Down-Down analysis. It follows that the linearization (7.24) about the Up-Up state has two positive eigenvalues and two negative eigenvalues. From the nomenclature above, the Up-Up state is an *index-2 saddle*.

From the above we find that the Down-Up and Up-Down states are index-1 saddles. Importantly, our work in Section 7.2 gives that in a neighborhood of these index-1 saddles there exist infinitely many UPOs. These UPOs are analogous to the Lyapunov orbits of the L_1 and L_2 Lagrange points of the PCR3BP. However, in the case of the double pendulum these UPOs may be easier to visualize. Precisely, the 'Down' arm of these index-1 saddles should be understood as being stable, while the 'Up' arm is unstable, both due to the force of gravity. Therefore, the UPOs near these Down-Up and Up-Down saddles have the 'Up' arm undergoing a slight wobble, while the 'Down' arm exhibits little variation since it is stable. This is demonstrated in Figure 7.5 where one can see a collection of these UPOs projected into both the $(\theta_1, \dot{\theta}_1)$ and $(\theta_2, \dot{\theta}_2)$ planes. In the following subsections we will work to numerically identify homoclinic and heteroclinic trajectories associated to these UPOs near the index-1 saddles of the double pendulum.

7.4.3 Tube Dynamics Near the Down-Up State

We begin by describing our process of identifying homoclinic trajectories to the UPOs near the Down-Up state of the double pendulum. We emphasize that although we refer to these orbits throughout as 'homoclinic' they are actually heteroclinic orbits in phase space. This potential source of confusion comes from the periodicity of the θ_1 and θ_2 components in (7.20). Precisely, a trajectory that connections the UPOs near the Down-Up state $(\theta_1, \theta_2, \omega_1, \omega_2) = (0, \pi, 0, 0)$ to those near another Down-Up state $(\theta_1, \theta_2, \omega_1, \omega_2) = (0, \pi \pm 2\pi, 0, 0)$ comes as a heteroclinic orbit in phase space if one does not quotient by the periodicity of the θ_1 and θ_2 components, but in physical space such a connection appears to return to where it started while having the second arm undergoing a full clockwise or counterclockwise rotation. As the motion of the double pendulum is best understood in physical space, we will hereby refer to trajectories that connect UPOs near any of the Down-Up states, $(0, \pi \pm 2\pi k, 0, 0)$, $k \in \mathbb{Z}$, as homoclinic. Finally, we comment that our numerical investigations have revealed that true homoclinic trajectories that asymptotically approach

one of the UPOs near $(0, \pi, 0, 0)$ do not exist, meaning that only 'physical' homoclinic trajectories described above can exist.

We can numerically obtain the UPOs near the Down-Up state by searching for symmetric periodic solutions with value of the Hamiltonian (7.14) slightly above that of the Down-Up state. The search for these periodic solutions can be posed as a root-finding problem constrained by the fact that we must remain within a Hamiltonian level set everywhere on the trajectory. This process is identical to that used in [439] to identify homoclinic trajectories in the PCR3BP and to implement this process numerically we use the Julia package $Zygote$ [469]. A demonstration of these techniques is included with the repository associated to this manuscript. With these UPOs, as displayed in Figure 7.5, we then use their Floquet spectrum to locally approximate the stable and unstable directions associated to the UPO. By simulating these approximate stable and unstable directions backward and forward in time, respectively, according to the ODE (7.20) we obtain accurate representations of the stable and unstable manifolds, i.e. the tubes, associated to each UPO.

To identify homoclinic trajectories we flow the unstable manifold forward in time until $\theta_2 = 2\pi$. Similarly, we flow the stable manifold backward in time until $\theta_2 = 0$. Notice that upon quotienting for the periodicity in θ_2 both the stable and unstable manifold have been simulated to the same region in physical space, given by the second arm hanging straight down. We use the hyperplane $\theta_2 = 2\pi k$ with $k \in \mathbb{Z}$ as our Poincaré section to identify homoclinic trajectories. We further note that the reversible symmetry of (7.20) guarantees that we could equivalently flow the unstable manifold forward in the direction that decreases the value of θ_2 until it reaches $\theta_2 = 0$ to meet the Poincaré section, and similarly for the stable manifold meeting $\theta_2 = 2\pi$. The tube structure of these stable and unstable manifolds are given in Figure 7.6(a) and their reversible counterparts are shown in Figure 7.6(b) for $\mathcal{H} = 0.2$, above the value of the Down-Up state at $\mathcal{H} = -0.1754$. Having the unstable manifold meet $\theta_2 = 2\pi$ represents the second arm moving clockwise, while having the unstable manifold meet $\theta_2 = 0$ represents the second arm moving

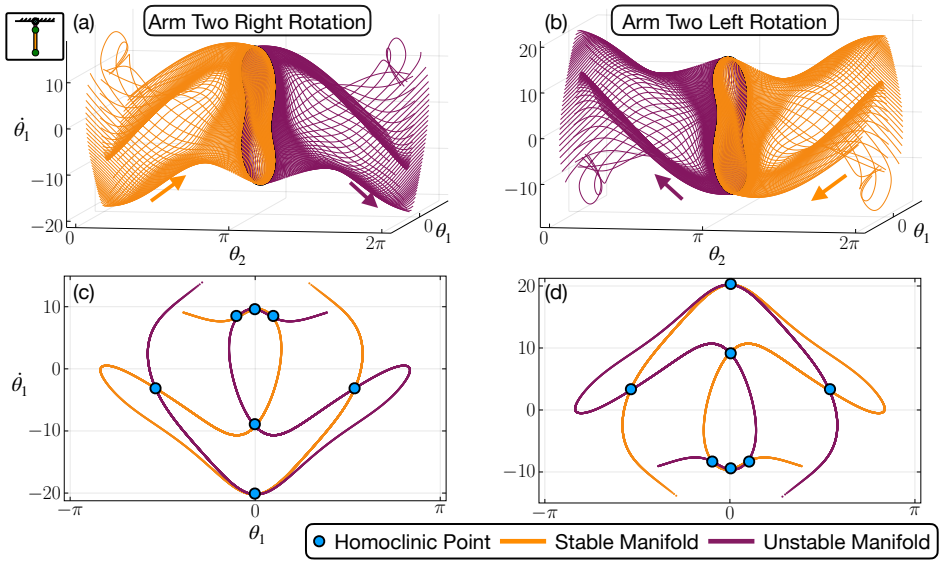


Figure 7.6: (a) The clockwise rotating portion of the tube structure of the stable and unstable manifolds associated to a UPO near the Down-Up state with Hamiltonian value $\mathcal{H} = 0.2$. In (b) we show the counterclockwise portion of the tube structure, related to (a) by applying the reverser (7.22). In (c) and (d) we present the intersection of the tubes with the Poincaré section $\theta_2 = 2\pi k$, $k \in \mathbb{Z}$. Intersections of the stable and unstable manifolds in the Poincaré section represent 'physical' homoclinic trajectories that connect the UPOs near the Down-Up state.

counterclockwise.

We can identify 'physical' homoclinic trajectories near the Down-Up state of the double pendulum using the Poincaré section $\theta_2 = 2\pi k$, $k \in \mathbb{Z}$. Indeed, Figure 7.6(c) and (d) plot the $(\theta_1, \dot{\theta}_1)$ plane corresponding to the tube structures presented in (a) and (b), respectively, of the same figure. Intersections of the stable and unstable manifolds in the Poincaré section represent the homoclinic orbits that are the focus of this section. Such intersections are homoclinic trajectories since θ_1, θ_2 and $\dot{\theta}_1$ are equal at these points, and it can be checked numerically that the Hamiltonian structure of system (7.20) confines the value of $\dot{\theta}_2$ to be the same here as well. As one can see, for this value of the Hamiltonian there are numerous intersections. Those with $\theta_1 = 0$ represent symmetric, or reversible, homoclinic orbits while those intersections with $\theta_1 \neq 0$ represent the asymmetric homoclinic orbits. As discussed

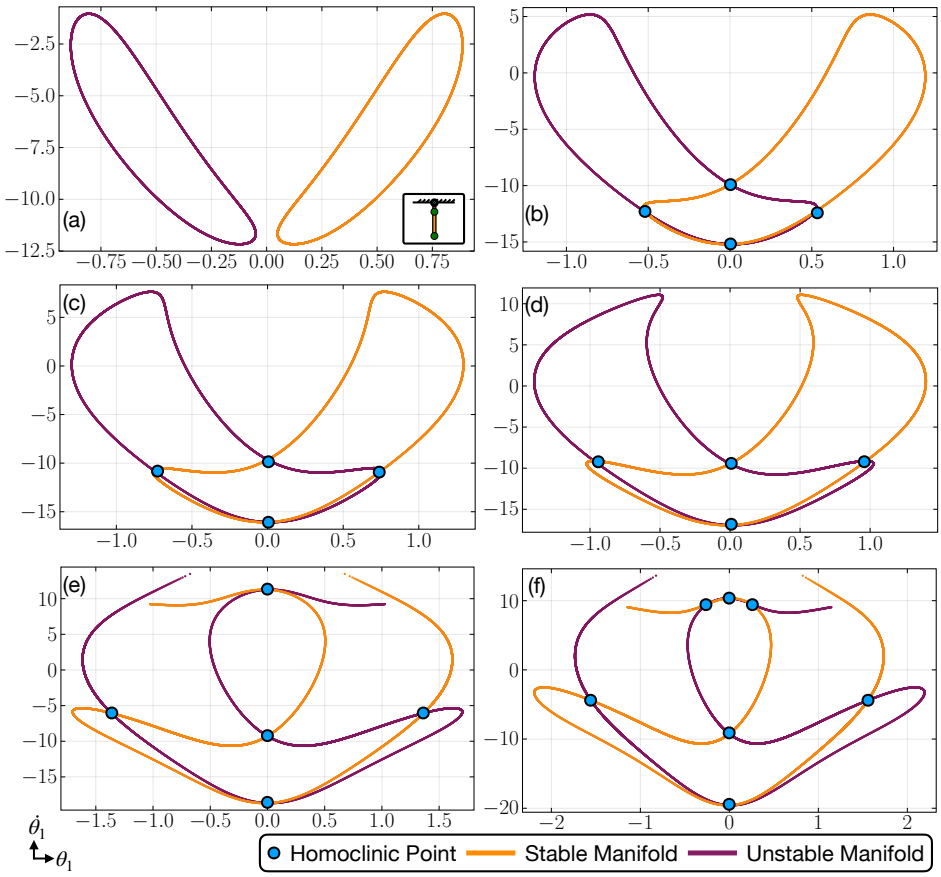


Figure 7.7: Increasing the energy of the double pendulum up from the energy of the Down-Up steady-state results in an increasing number of homoclinic trajectories of the UPOs near the Down-Up state. Energy values are (a) $\mathcal{H} = -0.147$, (b) $\mathcal{H} = -0.07$, (c) $\mathcal{H} = -0.034$, (d) $\mathcal{H} = 0.06$, (e) $\mathcal{H} = 0.102$, and (f) $\mathcal{H} = 0.157$. The Down-Up state has energy $\mathcal{H} = -0.1754$.

in § 7.4.1, the asymmetric orbits come in pairs, represented in the Poincaré section by the symmetry over the $\theta_1 = 0$ line. From our choice of the Poincaré section, all such homoclinic trajectories represent the double pendulum starting near the Down-Up state and having the second arm completing a full rotation before returning to a neighborhood of the Down-Up state.[1].

In Figure 7.7 we can see that as we increase the energy, given by the value of the

[1]To visualize the behavior of those homoclinic orbits presented in Fig. 7.6 in physical space see github.com/dynamicslab/Saddle-Mediated-Transport-of-Double-Pendulum

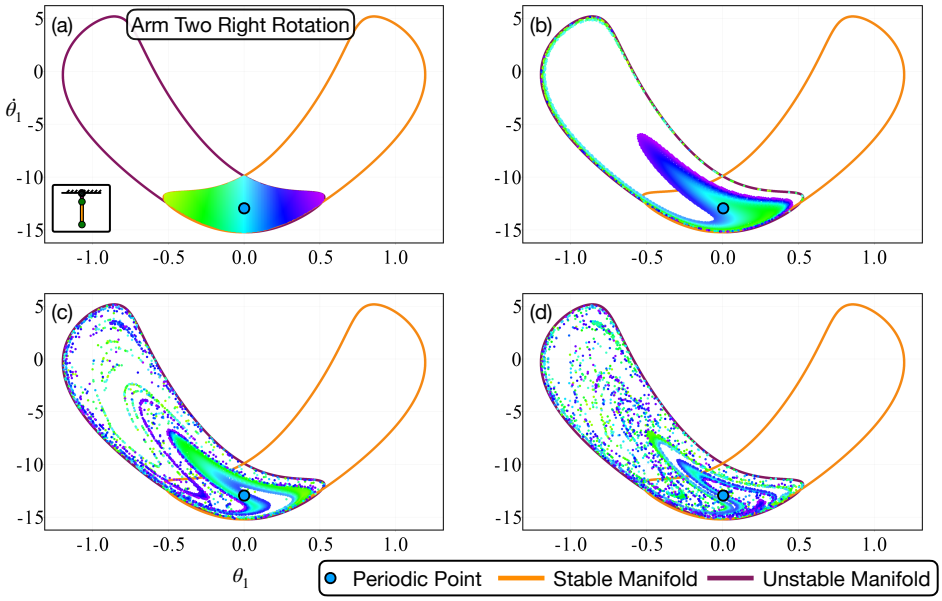


Figure 7.8: Trajectories that start inside of the unstable tube will remain inside of it for all forward time. This is demonstrated by flowing numerous points shown in (a) which lie inside both the stable and unstable tubes of a UPO near the Down-Up state with energy $\mathcal{H} = -0.06985$ forward until they intersect the Poincaré section again at (b) $\theta_2 = 4\pi$, (c) $\theta_2 = 6\pi$, and (d) $\theta_2 = 8\pi$. Flowing these same points backward in time produces results that are mirrored over $\theta_1 = 0$, likely resulting in a fractal set that remains inside both the stable and unstable tubes for all forward and backward time. The location of a symmetric periodic orbit that remains inside both tubes for all times is also plotted.

Hamiltonian (7.14), from that of the Down-Up state results in a steadily increasing number of homoclinic trajectories. These homoclinic orbits arise via saddle-node and symmetry-breaking pitchfork bifurcations as the energy is increased, coming from further manifold intersections. For relatively large values of the energy, as is in panels (e) and (f), flowing some parts of stable and unstable manifolds towards the Poincaré section takes a long time. The result is that some of our intersections of the stable and unstable manifolds with the Poincaré section do not present themselves as closed loops. We expect that simulating the tubes for a significantly longer time will complete their intersections with the Poincaré section, but this would require significant computational time and is not necessary for our exposition here.

The fact that these stable and unstable tubes intersect means that the system exhibits macroscopic transport mechanisms that allow one to traverse large distances in phase space, eventually returning to the region where it started. Precisely, trajectories that are initialized inside of an unstable tube will remain inside of it for all forward time, thus repeatedly coming back to the Poincaré section $\theta_2 = 2\pi k$ and intersecting within the closed curve created by the intersection with the unstable tube. This is demonstrated in Figure 7.8 where we simulate numerous trajectories with initial conditions from the Poincaré section that lie in the intersection of the stable and unstable tubes for energy $\mathcal{H} = -0.06985$. After each pass from the Poincaré section up to the UPO near the Down-Up state and back one sees that these trajectories intersect the Poincaré section inside the closed curve created by the unstable tube, while spreading out over the interior of the tube. One can achieve a similar phenomenon by simulating these initial conditions backward in time, forcing them to remain inside of the stable tube forever. Visualizing this can be achieved by simply mirroring the forward iterates in Figure 7.8 over $\theta_1 = 0$. Thus, it appears that those trajectories which remain inside both tubes for all forward and backward time becomes a fractal set which we suspect forms a horseshoe set [470].

This apparent horseshoe structure means that there exist a plethora of bounded solutions that return infinitely often to a neighborhood of the UPO near the Down-Up state. As an example, we have identified a simple symmetric periodic orbit that in physical space has the second arm of the double pendulum completing a full 2π rotation over each period. The intersection of this periodic orbit with the Poincaré section is plotted in Figure 7.8. Interestingly, it lies almost perfectly between the location of the two symmetric homoclinic orbits that form the boundary of the tubes restricted to $\theta_1 = 0$. Although we do not have an explanation for this, we do not believe it to be a coincidence and expect it to be related to the precise horseshoe structure generated by the intersection of the interiors of the stable and unstable tubes.

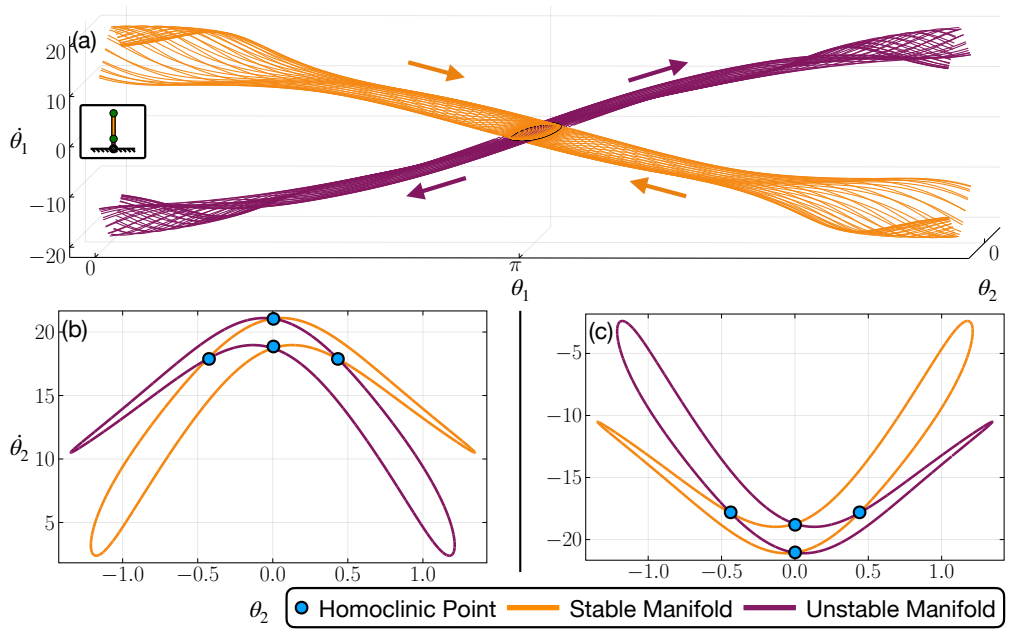


Figure 7.9: Panel (a) presents the stable and unstable tubes of a UPO near the Up-Down steady-state in the variables $(\theta_1, \theta_2, \dot{\theta}_1)$ with energy $\mathcal{H} = 0.2$. In (b) and (c) we present the intersection of the tubes with the Poincaré section $\theta_1 = 2\pi k$, $k \in \mathbb{Z}$. Intersections of the stable and unstable manifolds in the Poincaré section represent 'physical' homoclinic trajectories that connect the UPOs near the Up-Down state.

7.4.4 Tube Dynamics Near the Up-Down State

The process of identifying homoclinic trajectories to the UPOs near the Up-Down state is similar to what was done in the previous subsection. Again, we were not able to identify homoclinic trajectories in phase space but have sought to identify the 'physical' homoclinic trajectories that connect UPOs near the Up-Down state after accounting for the periodicity of the θ_1 and θ_2 variables. For the UPOs near the Up-Down state we find that these physical homoclinic trajectories have the first arm making a full clockwise or counterclockwise rotation before returning to where they started. In phase space this means that θ_1 increases or decreases by exactly 2π over the course of the homoclinic trajectory.

We can identify the UPOs near the Up-Down state and use their Floquet spectrum to follow the stable and unstable tubes backward and forward in time, respectively. We

flow these stable and unstable tubes associated to these UPOs according to (7.20) until they meet the Poincaré section $\theta_1 = 2\pi k$, $k \in \mathbb{Z}$. In Figure 7.9(a) we present the stable and unstable manifolds of a UPO near the Up-Down state with energy $\mathcal{H} = 0.2$. The resulting intersections with the Poincaré section are further plotted in panels (b) and (c), for which the former represents clockwise rotations of the first arm, while the latter represents counterclockwise rotations. Again we are able to identify multiple symmetric and asymmetric homoclinic orbits that enable macroscopic transport throughout the phase space of the double pendulum[2]. As with the UPOs near the Down-Up state, we find that increasing the energy above that of the Up-Down state results in sequences of saddle-node and symmetry-breaking pitchfork bifurcations to more and more homoclinic orbits. We do not present a figure with these results for brevity.

As with the tubes associated to UPOs near the Down-Up state, we find a fractal/horseshoe structure of trajectories that remain inside the tubes associated to UPOs near the Up-Down state for all forward and backward time. This structure is illustrated in Figure 7.10 for energy $\mathcal{H} = 0.2$, which is analogous to Figure 7.8 presented previously. We again exploit this complex structure to identify an exemplary symmetric periodic orbit that represents the first arm making a full 2π rotation in physical space. These symmetric orbits intersect the Poincaré section at $\theta_2 = 0$ and again we find this point of intersection lies almost perfectly in the center of the intersections of two symmetric homoclinic orbits.

7.4.5 Heteroclinic Transitions

The approach to identifying heteroclinic connections between neighborhoods of the Down-Up and Up-Down states is similar to that outlined above for identifying homoclinic trajectories. We fix the energy and flow the unstable manifold of the a UPO near the Down-Up state forward in time until it reaches the Poincaré section $\theta_1 = \theta_2$ in phase space. We then flow the stable manifold of a UPO with the same energy near the Up-Down

[2]To visualize the behavior of those homoclinic orbits presented in Fig. 7.9 in physical space see github.com/dynamicslab/Saddle-Mediated-Transport-of-Double-Pendulum.

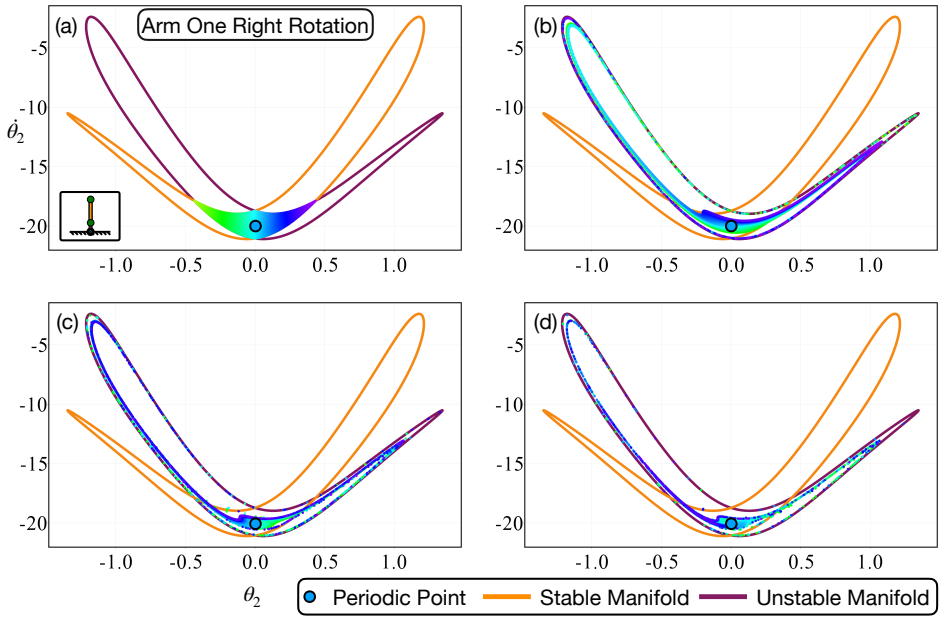


Figure 7.10: The apparent fractal structure generated by those trajectories that remain inside of the unstable tube of a UPO near the Up-Down state with energy $\mathcal{H} = 0.2$. As in Figure 7.8 we flow numerous initial conditions shown in (a) forward in time until they intersect the Poincaré section for a (b) first, (c) second, and (d) third time. The location of a symmetric periodic orbit that remains inside both tubes for all times is also plotted.

state backwards in time until it also reaches the Poincaré section $\theta_1 = \theta_2$. Intersections of these stable and unstable manifolds in the Poincaré section therefore represent the desired heteroclinic orbits that transport one through phase space between neighborhoods of index-1 saddles. The process we have just described allows one to obtain orbits from near the Down-Up state to near the Up-Down state, although applying the reverser (7.22) reverses the direction of these orbits and therefore gives orbits that originate near the Up-Down state and move to a neighborhood of the Down-Up state as well.

Our results are illustrated in Figure 7.11. Panel (a) demonstrates the unstable manifold of a UPO near the Down-Up state moving toward the Poincaré section, while panel (c) demonstrates the time reversed flow of the stable manifold of the same UPO moving backward in time toward the Poincaré section. Panels (b) and (d) show the intersection of the

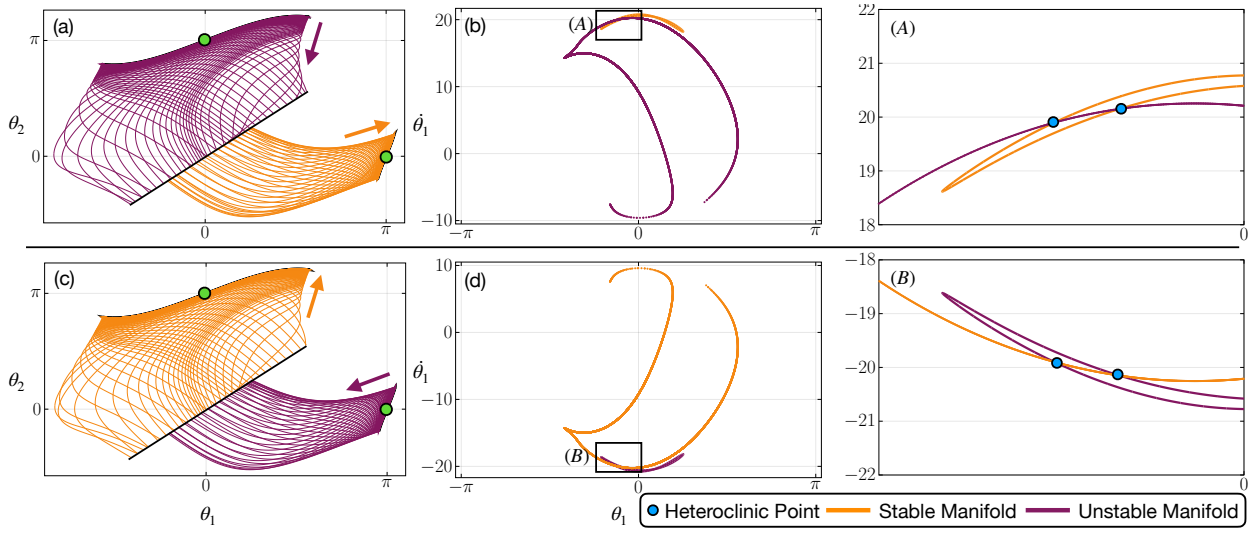


Figure 7.11: Heteroclinic connections between UPOs in the neighborhoods of the Down-Up and Up-Down states are found by inspecting where their tubes meet the Poincaré section $\theta_1 = \theta_2$. (a) and (c) show the tubes flowing towards the section in forward and backward time for $\mathcal{H} = 0.2$. Panels (b) and (d) demonstrate the intersection of the tubes with the Poincaré section and panels (A) and (B) provide a zoom of the intersection of these tubes, representing heteroclinic trajectories.

UPO tubes with the Poincaré section for panels (a) and (c), respectively. We further present a zoom of the Poincaré section near the intersections, illustrating two distinct heteroclinic connections. These heteroclinic orbits are plotted in $(\theta_1, \theta_2, \dot{\theta}_1)$ space in Figure 7.12 with their asymptotic UPOs shown in solid black. Our numerical experiments have shown that such heteroclinic connections exist for $\mathcal{H} \geq 0.1754$. Like the homoclinic orbits of the previous subsections, the existence of these heteroclinic orbits demonstrates a transport mechanism that allows one to move between neighbourhoods of the index-1 saddles in phase space, thus allowing saddle-mediated transport of the double pendulum.

7.5 Conclusion

In this Chapter, we performed a detailed numerical analysis of the double pendulum following the work of [437, 439, 467]. We find a striking resemblance between the tube

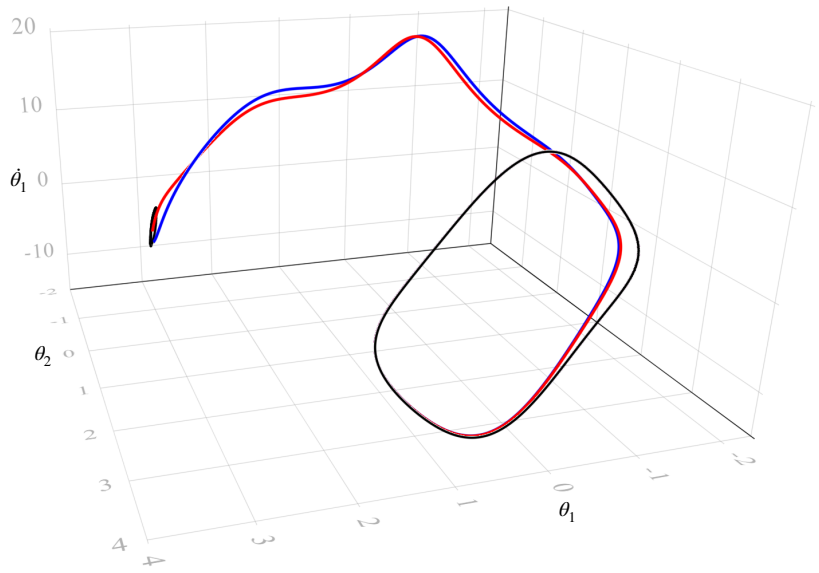


Figure 7.12: A visualization in $(\theta_1, \theta_2, \dot{\theta}_1)$ space of the two distinct heteroclinic orbits (red and blue) found as intersections in the Poincaré section $\theta_1 = \theta_2$ in Figure 7.11. The black closed loops represent the asymptotic UPOs near each of the index-1 saddles.

dynamics of the PCR3BP and double pendulum since both of them have two index-1 saddle points. We showed there exists a family of Lyapunov orbits near the saddle points of the double pendulum. These orbits allow us to flow the stable and unstable manifold of the double pendulum. The intersections of the stable and unstable manifolds facilitate the identification of the double pendulum's homoclinic and heteroclinic orbits. The homoclinic orbits of the double pendulum allow the pendulum to travel in the phase space and return to the same Lyapunov orbits, while the heteroclinic orbits transport the double pendulum from one saddle point to the other. Thus, we can achieve saddle-mediated transport of the double pendulum by using the homoclinic and heteroclinic orbits.

Our numerical analysis shows the double pendulum can be used as a low-stakes and relatively low-cost tabletop experiment to explore the saddle-mediated transport put forth for the PCR3BP in [437]. This is extremely exciting since we can now have a double pendulum system to test out the saddle-mediated space travel controller for a fraction of the cost of building a satellite or space shuttle. This helps researchers quickly check the

performance of their control algorithm before implementing them on the actual satellite. Thus, our future work includes using the experimental system in Chapter 6 to validate the saddle-mediated transport idea shown in this Chapter.

Chapter 8
CONCLUSION

In this thesis, we presented several topics in the field of the dynamical system. First, we introduced some of the techniques and tools available in the data-driven modeling of the nonlinear dynamical system. Then, in Chapter 2, we specifically focused on the SINDy algorithm and did a detailed literature review on it. We showed multiple sparsity-promoting techniques could be used in the SINDy framework and showed that SINDy has many variants and applications. Our study on the related work of SINDy motivated our work on robustifying the SINDy algorithm, resulting in SINDy-PI and modified SINDy algorithms.

In Chapter 3, we introduced SINDy-PI. The SINDy-PI is a robust variant of the SINDy algorithm that identifies implicit dynamics and rational nonlinearities. It improves the robustness of the SINDy algorithm by overcoming the sensitivity of the previous implicit-SINDy approach, which is based on a null-space calculation. The work of SINDy-PI extended the applicability of the SINDy algorithm and made the identification of physical laws possible. The second variant we developed is modified SINDy, and it is introduced in Chapter 4. This algorithm leverages automatic differentiation and sparse regression to denoise the time-series data and learn noise probability distribution simultaneously. It can also identify the underlying parsimonious dynamical system responsible for generating the time-series data. This algorithm is highly robust, allowing top-of-the-line denoising abilities in the SINDy framework. When integrated with the discrepancy modeling framework, modified SINDy further improves its applicability, as shown in Chapter 5.

When developing those extensions to the SINDy algorithm, we realized it is vital to have an experimental system to test our proposed algorithms. This motivates us to create

a fully open-source multi-link pendulum on the cart system shown in Cahpter 6. Our proposed system is flexible and allows the learning of chaotic behavior, testing data-driven modeling techniques, and validating different control algorithms. We detailed our building procedures with the freely available CAD files and data set collected from the experimental setup. We believe our system will become a valuable resource for the machine learning community. One more reason we developed this experimental system is to study the saddle-mediated transport of the double pendulum shown in Chapter 7. Chapter 7 offers a detailed analysis of the double pendulum's stable and unstable manifold and its UPOs. Our study shows double pendulum is a great tabletop analog system of the PCR3BP since they both have two index-1 saddle points. The existence of two index-1 saddle points made the saddle-mediated transport of the double pendulum possible by utilizing its homoclinic and heteroclinic orbits.

There are several future directions for our work. 1) It is interesting to see that in the constrained optimization problem of SINDy-PI, the final solution highly resembles the adjacency matrix in the network theory. Thus, it would be interesting to research in this direction and see the connection between SINDy-PI and the adjacency matrix. 2) It is necessary to improve the speed of the modified SINDy algorithm and its applicability. Currently, the modified SINDy algorithm is slow due to the ADAM optimizer. We also need to consider identifying the rational, implicit, or controlled system in the modified SINDy framework. 3) When it comes to the discrepancy modeling framework, more works need to be done to study how to incorporate physical law constraints into the discrepancy model. 4) As for the hardware setup of the pendulum system, we want to explore the idea of cloud experiments and physically realize it. 5) Finally, we want to use the numerical analysis result on the double pendulum and experimentally validate the possibility of the saddle-mediated transport using the experimental system we built.

BIBLIOGRAPHY

[1] R.J. Seeger and D. Haar. *Men of Physics: Galileo Galilei, His Life and His Works*. Commonwealth and international library: Selected readings in physics. Elsevier Science, 2016.

[2] Eric J. Aiton. How Kepler discovered the elliptical orbit. *The Mathematical Gazette*, 59(410):250–260, 1975.

[3] Isaac Newton, Daniel Bernoulli, Colin MacLaurin, and Leonhard Euler. *Philosophiae naturalis principia mathematica*, volume 1. excudit G. Brookman; impensis TT et J. Tegg, Londini, 1833.

[4] Steve L. Brunton, J. Nathan Kutz, and Joshua L. Proctor. Data-driven discovery of governing physical laws. https://sinews.siam.org/Details-Page/data-driven-discovery-of-governing-physical-laws, January 2017.

[5] R.M. Henig. *The Monk in the Garden: The Lost and Found Genius of Gregor Mendel, the Father of Genetics*. Collection David M.-Lank. Houghton Mifflin, 2000.

[6] Pat Langley, Gary L. Bradshaw, and Herbert A. Simon. BACON.5: The discovery of conservation laws. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 1*, IJCAI'81, pages 121–126, San Francisco, CA, USA, 1981. Morgan Kaufmann Publishers Inc.

[7] Knut Graichen, Michael Treuer, and Michael Zeitz. Swing-up of the double pendulum on a cart by feedforward and feedback control with experimental validation. *Automatica*, 43(1):63–71, 2007.

[8] Yang Liang and Brian Feeny. Parametric identification of chaotic base-excited double pendulum experiment. *Nonlinear Dynamics*, 52:181–197, 01 2004.

[9] Ivan Koryakovskiy, Manuel Kudruss, Heike Vallery, Robert Babuška, and Wouter Caarls. Model-plant mismatch compensation using reinforcement learning. *IEEE Robotics and Automation Letters*, 3(3):2471–2477, 2018.

[10] P.K. Kundu, I.M. Cohen, and D.R. Dowling. *Fluid Mechanics*. Science Direct e-books. Elsevier Science, 2012.

[11] J. Nathan Kutz. *Data-Driven Modeling & Scientific Computation: Methods for Complex Systems & Big Data*. Oxford University Press, Inc., New York, NY, USA, 2013.

[12] Samuel H Rudy, Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Data-driven discovery of partial differential equations. *Science Advances*, 3(4):e1602614, 2017.

[13] Josh Bongard and Hod Lipson. Automated reverse engineering of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 104(24):9943–9948, 2007.

[14] Michael Schmidt and Hod Lipson. Distilling free-form natural laws from experimental data. *Science*, 324(5923):81–85, 2009.

[15] Kadierdan Kaheman, Steven L Brunton, and J Nathan Kutz. Automatic differentiation to simultaneously identify nonlinear dynamics and extract noise probability distributions from data. *Machine Learning: Science and Technology*, 3(1):015031, 2022.

[16] Firdous Ul Rashid. A review on DENDRAL expert system. *International Journal for Research in Applied Science & Engineering Technology*, 6(2), 2018.

[17] Robert K. Lindsay, Bruce G. Buchanan, Edward A. Feigenbaum, and Joshua Lederberg. DENDRAL: A case study of the first expert system for scientific hypothesis formation. *Artificial Intelligence*, 61(2):209 – 261, 1993.

[18] Edward A. Feigenbaum and Bruce G. Buchanan. DENDRAL and Meta-DENDRAL: Roots of knowledge systems and expert system applications. *Artificial Intelligence*, 59(1):233 – 240, 1993.

[19] Pat Langley. BACON: A production system that discovers empirical laws. In *IJCAI*, 1977.

[20] Pat Langley. Rediscovering physics with BACON.3. In *Proceedings of the 6th International Joint Conference on Artificial Intelligence - Volume 1*, IJCAI'79, pages 505–507, San Francisco, CA, USA, 1979. Morgan Kaufmann Publishers Inc.

[21] Pat Langley. Data-driven discovery of physical laws. *Cognitive Science*, 5(1):31–54, 1981.

[22] PAT LANGLEY. The computational support of scientific discovery. *International Journal of Human-Computer Studies*, 53(3):393 – 410, 2000.

[23] A. L. Samuel. Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, 44(1.2):206–226, Jan 2000.

[24] J.H. Holland. *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence*. University of Michigan Press, 1975.

[25] John H. Holland. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. MIT Press, Cambridge, MA, USA, 1992.

[26] D.S. Broomhead and Gregory P. King. Extracting qualitative dynamics from experimental data. *Physica D: Nonlinear Phenomena*, 20(2):217 – 236, 1986.

[27] Floris Takens. Detecting strange attractors in turbulence. In David Rand and Lai-Sang Young, editors, *Dynamical Systems and Turbulence, Warwick 1980*, pages 366–381, Berlin, Heidelberg, 1981. Springer Berlin Heidelberg.

[28] James P. Crutchfield and Bruce S. Mcnamara. Equations of motion from a data series. *Complex Systems*, page 452, 1987.

[29] Nichael Lynn Cramer. A representation for the adaptive generation of simple sequential programs. In *Proceedings of the 1st International Conference on Genetic Algorithms*, pages 183–187, Hillsdale, NJ, USA, 1985. L. Erlbaum Associates Inc.

[30] John R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, Cambridge, MA, USA, 1992.

[31] Ivan Zelinka, Zuzana Kominkova Oplatkova, and Lars Nolle. Analytic programming–Symbolic regression by means of arbitrary evolutionary algorithms. *J. of SIMULATION*, 6:1473–8031, 08 2005.

[32] Mark Willis, Hugo Hiden, Peter Marenbach, Ben McKay, and Gary A. Montague. Genetic programming: An introduction and survey of applications. In Ali Zalzala, editor, *Second International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications, GALESIA*, pages 314–319, University of Strathclyde, Glasgow, UK, 1-4 September 1997. Institution of Electrical Engineers.

[33] S Forrest. Genetic algorithms: Principles of natural selection applied to computation. *Science*, 261(5123):872–878, 1993.

[34] Wolfgang Banzhaf, Frank D. Francone, Robert E. Keller, and Peter Nordin. *Genetic Programming: An Introduction on the Automatic Evolution of Computer Programs and Its Applications*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1998.

[35] S. Mashohor, J. R. Evans, and T. Arslan. Elitist selection schemes for genetic algorithm based printed circuit board inspection system. In *2005 IEEE Congress on Evolutionary Computation*, volume 2, pages 974–978 Vol. 2, Sep. 2005.

[36] J.R Koza. A genetic approach to the truck backer upper problem and the inter-twined spiral problem. In *[Proceedings 1992] IJCNN International Joint Conference on Neural Networks*, volume 4, pages 310–318 vol.4. IEEE, 1992.

[37] Conor Ryan and Michael O'Neill. Grammatical evolution: A steady state approach. *Late Breaking Papers, Genetic Programming*, 1998:180–185, 1998.

[38] Q. Chen, B. Xue, and M. Zhang. Improving generalization of genetic programming for symbolic regression with angle-driven geometric semantic operators. *IEEE Transactions on Evolutionary Computation*, 23(3):488–502, June 2019.

[39] T. Soule and J. A. Foster. Removal bias: A new cause of code growth in tree based evolutionary programming. In *1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98TH8360)*, pages 781–786, May 1998.

[40] Donald S. Burke, Kenneth A. De Jong, John J. Grefenstette, Connie Loggia Ramsey, and Annie S. Wu. Putting more genetics into genetic algorithms. *Evol. Comput.*, 6(4):387–410, December 1998.

[41] Jeffrey K. Bassett and Kenneth A. De Jong. Evolving behaviors for cooperating agents. In *Proceedings of the 12th International Symposium on Foundations of Intelligent Systems*, ISMIS '00, pages 157–165, Berlin, Heidelberg, 2000. Springer-Verlag.

[42] Byoung-Tak Zhang and Heinz Mühlenbein. Balancing accuracy and parsimony in genetic programming. *Evol. Comput.*, 3(1):17–38, March 1995.

[43] Sean Luke and Liviu Panait. Fighting bloat with nonparametric parsimony pressure. In *Proceedings of the 7th International Conference on Parallel Problem Solving from Nature*, PPSN VII, pages 411–421, London, UK, UK, 2002. Springer-Verlag.

[44] Edwin D. De Jong and Jordan B. Pollack. Multi-objective methods for tree size control. *Genetic Programming and Evolvable Machines*, 4(3):211–233, September 2003.

[45] Christian Gagné, Marc Parizeau, and Marc Dubreuil. A robust master-slave distribution architecture for evolutionary computations. In *in Late Breaking Papers of GECCO 2003*, 2003.

[46] Francisco Fernández, Giandomenico Spezzano, Marco Tomassini, and Leonardo Vanneschi. *Parallel Genetic Programming*, chapter 6, pages 127–153. John Wiley & Sons, Ltd, 2005.

[47] M. D. Schmidt and H. Lipson. Coevolution of fitness predictors. *Trans. Evol. Comp*, 12(6):736–749, December 2008.

[48] Michael O'Neill, Leonardo Vanneschi, Steven Gustafson, and Wolfgang Banzhaf. Open issues in genetic programming. *Genetic Programming and Evolvable Machines*, 11:339–363, 09 2010.

[49] Daniel Hills, Adrian Grütter, and Jonathan Hudson. An algorithm for discovering Lagrangians automatically from data. *PeerJ Computer Science*, 1, 06 2015.

[50] Seyedali Mirjalili. Genetic algorithm. In *Evolutionary algorithms and neural networks*, pages 43–55. Springer, 2019.

[51] Manoj Kumar, Mohammad Husain, Naveen Upreti, and Deepti Gupta. Genetic algorithm: Review and application. *Available at SSRN 3529843*, 2010.

[52] Sourabh Katoch, Sumit Singh Chauhan, and Vijay Kumar. A review on genetic

algorithm: Past, present, and future. *Multimedia Tools and Applications*, 80(5):8091–8126, 2021.

[53] L Haldurai, T Madhubala, and R Rajalakshmi. A study on genetic algorithm and its applications. *International journal of computer sciences and Engineering*, 4(10):139, 2016.

[54] Annu Lambora, Kunal Gupta, and Kriti Chopra. Genetic algorithm-A literature review. In *2019 international conference on machine learning, big data, cloud and parallel computing (COMITCon)*, pages 380–384. IEEE, 2019.

[55] Seyedali Mirjalili, Jin Song Dong, Ali Safa Sadiq, and Hossam Faris. Genetic algorithm: Theory, literature review, and application in image reconstruction. *Nature-inspired optimizers*, pages 69–85, 2020.

[56] Lennart Ljung. Perspectives on system identification. *Annual Reviews in Control*, 34(1):1–12, 2010.

[57] Benjamin Peherstorfer and Karen Willcox. Data-driven operator inference for nonintrusive projection-based model reduction. *Computer Methods in Applied Mechanics and Engineering*, 306:196–215, 2016.

[58] Nihar Sawant, Boris Kramer, and Benjamin Peherstorfer. Physics-informed regularization and structure preservation for learning stable reduced models from data with operator inference. *arXiv preprint arXiv:2107.02597*, 2021.

[59] S. L. Brunton and J. N. Kutz. *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*. Cambridge University Press, 2019.

[60] L. Ljung. State of the art in linear system identification: Time and frequency domain methods. In *Proceedings of the 2004 American Control Conference*, volume 1, pages 650–660 vol.1, 2004.

[61] Joannes Schoukens, Rik Pintelon, and Yves Rolain. Time domain identification, frequency domain identification. Equivalencies! Differences? In *Proceedings of the 2004 American Control Conference*, volume 1, pages 661–666. IEEE, 2004.

[62] Hugues Garnier and Peter Young. Time-domain approaches to continuous-time model identification of dynamical systems from sampled data. In *Proceedings of the 2004 American control conference*, volume 1, pages 667–672. IEEE, 2004.

[63] Hugues Garnier, MICHEL Mensler, and ALAIN Richard. Continuous-time model identification from sampled data: Implementation issues and performance evaluation. *International journal of Control*, 76(13):1337–1357, 2003.

[64] Tomas McKelvey. Subspace methods for frequency domain data. In *Proceedings of the 2004 American control conference*, volume 1, pages 673–678. IEEE, 2004.

[65] H Unbehauen and GP Rao. A review of identification in continuous-time systems. *Annual reviews in Control*, 22:145–171, 1998.

[66] Karel J Keesman and Karel J Keesman. *System identification: An introduction*, volume 2. Springer, 2011.

[67] Christiaan Heij, André CM Ran, and Frederik van Schagen. *Introduction to mathematical systems theory: Linear systems, identification and control*. Springer Science & Business Media, 2006.

[68] Stephen A Billings. *Nonlinear system identification: NARMAX methods in the time, frequency, and spatio-temporal domains*. John Wiley & Sons, 2013.

[69] Hirotugu Akaike. Fitting autoregressive models for prediction. *Annals of the institute of Statistical Mathematics*, 21(1):243–247, 1969.

[70] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Machine learning

of linear differential equations using Gaussian processes. *Journal of Computational Physics*, 348:683–693, 2017.

[71] Matthias Seeger. Gaussian processes for machine learning. *International journal of neural systems*, 14(02):69–106, 2004.

[72] Motonobu Kanagawa, Philipp Hennig, Dino Sejdinovic, and Bharath K Sriperumbudur. Gaussian processes and kernel methods: A review on connections and equivalences. *arXiv preprint arXiv:1807.02582*, 2018.

[73] Maziar Raissi and George Em Karniadakis. Hidden physics models: Machine learning of nonlinear partial differential equations. *Journal of Computational Physics*, 357:125–141, 2018.

[74] Juš Kocijan, Agathe Girard, Blaž Banko, and Roderick Murray-Smith. Dynamic systems identification with Gaussian processes. *Mathematical and Computer Modelling of Dynamical Systems*, 11(4):411–424, 2005.

[75] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Numerical Gaussian processes for time-dependent and nonlinear partial differential equations. *SIAM Journal on Scientific Computing*, 40(1):A172–A198, 2018.

[76] Liu Yang, Dongkun Zhang, and George Em Karniadakis. Physics-informed generative adversarial networks for stochastic differential equations. *arXiv preprint arXiv:1811.02033*, 2018.

[77] Christoph Wehmeyer and Frank Noé. Time-lagged autoencoders: Deep learning of slow collective variables for molecular kinetics. *The Journal of Chemical Physics*, 148(241703):1–9, 2018.

[78] Andreas Mardt, Luca Pasquali, Hao Wu, and Frank Noé. VAMPnets: Deep learning of molecular kinetics. *Nature Communications*, 9(5), 2018.

[79] Pantelis R Vlachas, Wonmin Byeon, Zhong Y Wan, Themistoklis P Sapsis, and Petros Koumoutsakos. Data-driven forecasting of high-dimensional chaotic systems with long short-term memory networks. *Proc. R. Soc. A*, 474(2213):20170844, 2018.

[80] Jaideep Pathak, Brian Hunt, Michelle Girvan, Zhixin Lu, and Edward Ott. Model-free prediction of large spatiotemporally chaotic systems from data: A Reservoir computing approach. *Physical review letters*, 120(2):024102, 2018.

[81] Lu Lu, Xuhui Meng, Zhiping Mao, and George Em Karniadakis. DeepXDE: A deep learning library for solving differential equations. *SIAM Review*, 63(1):208–228, 2021.

[82] M Raissi, P Perdikaris, and GE Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

[83] K. Champion, B. Lusch, J. Nathan Kutz, and Steven L. Brunton. Data-driven discovery of coordinates and governing equations. *Proceedings of the National Academy of Sciences*, 116(45):22445–22451, 2019.

[84] Maziar Raissi, Alireza Yazdani, and George Em Karniadakis. Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations. *Science*, 367(6481):1026–1030, 2020.

[85] Marko Budišić, Ryan Mohr, and Igor Mezić. Applied Koopmanism. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 22(4):047510, 2012.

[86] Igor Mezić. Analysis of fluid flows via spectral properties of the Koopman operator. *Annual Review of Fluid Mechanics*, 45:357–378, 2013.

[87] Matthew Williams, Ioannis Kevrekidis, and Clarence Rowley. A data-driven approximation of the Koopman operator: Extending dynamic mode decomposition. *Journal of Nonlinear Science*, 25, 08 2014.

[88] Stefan Klus, Feliks Nüske, Péter Koltai, Hao Wu, Ioannis Kevrekidis, Christof Schütte, and Frank Noé. Data-driven model reduction and transfer operator approximation. *Journal of Nonlinear Science*, 2018.

[89] Stephen A Billings and Daniel Coca. Identification of NARMAX and related models. *Research Report-University of Sheffield Department of Automatic Control and System Engineering*, 2001.

[90] Yuanlin Gu, Hua-Liang Wei, Michael A Balikhin, Richard J Boynton, and Simon N Walker. Machine learning enhanced NARMAX model for Dst index forecasting. In *2019 25th International Conference on Automation and Computing (ICAC)*, pages 1–6. IEEE, 2019.

[91] Michael A Balikhin, Richard J Boynton, Simon N Walker, Joe E Borovsky, Stephen A Billings, and Hua-Liang Wei. Using the NARMAX approach to model the evolution of energetic electrons fluxes at geostationary orbit. *Geophysical Research Letters*, 38(18), 2011.

[92] N Chiras, C Evans, and D Rees. Global nonlinear modeling of gas turbine dynamics using NARMAX structures. *J. Eng. Gas Turbines Power*, 124(4):817–826, 2002.

[93] Yewei Yu, Chen Zhang, and Miaolei Zhou. NARMAX model-based hysteresis modeling of magnetic shape memory alloy actuators. *IEEE Transactions on Nanotechnology*, 19:1–4, 2019.

[94] Richard J Boynton, Michael A Balikhin, Stephen A Billings, Hua-Liang Wei, and Natalia Ganushkina. Using the NARMAX OLS-ERR algorithm to obtain the most influential coupling functions that affect the evolution of the magnetosphere. *Journal of Geophysical Research: Space Physics*, 116(A5), 2011.

[95] Eric HK Fung, YK Wong, HF Ho, and Marc P Mignolet. Modelling and prediction

of machining errors using ARMAX and NARMAX structures. *Applied Mathematical Modelling*, 27(8):611–627, 2003.

[96] Zhihua Deng, Qihong Chen, Liyan Zhang, and Zhichao Fu. Data driven NARMAX modeling for PEMFC air compressor. *International Journal of Hydrogen Energy*, 45(39):20321–20328, 2020.

[97] Richard Boynton, Michael Balikhin, Hua-Liang Wei, and Zi-Qiang Lang. Applications of NARMAX in space weather. *Machine learning techniques for space weather*, pages 203–236, 2018.

[98] Eric Schulz, Maarten Speekenbrink, and Andreas Krause. A tutorial on Gaussian process regression: Modelling, exploring, and exploiting functions. *Journal of Mathematical Psychology*, 85:1–16, 2018.

[99] James Hensman, Nicolo Fusi, and Neil D Lawrence. Gaussian processes for big data. *arXiv preprint arXiv:1309.6835*, 2013.

[100] Haitao Liu, Yew-Soon Ong, Xiaobo Shen, and Jianfei Cai. When Gaussian process meets big data: A review of scalable GPs. *IEEE transactions on neural networks and learning systems*, 31(11):4405–4423, 2020.

[101] Juš Kocijan, Roderick Murray-Smith, Carl Edward Rasmussen, and Agathe Girard. Gaussian process model based predictive control. In *Proceedings of the 2004 American control conference*, volume 3, pages 2214–2219. IEEE, 2004.

[102] Juš Kocijan. *Modelling and control of dynamic systems using Gaussian process models*. Springer, 2016.

[103] Bojan Likar and Juš Kocijan. Predictive control of a gas–liquid separation plant based on a Gaussian process model. *Computers & Chemical Engineering*, 31(3):142–152, 2007.

[104] MP. Deisenroth and CE. Rasmussen. PILCO: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on Machine Learning, ICML 2011*, pages 465–472. Omnipress, 2011.

[105] TJ Rogers, P Gardner, N Dervilis, K Worden, AE Maguire, E Papatheou, and EJ Cross. Probabilistic modelling of wind turbine power curves with application of heteroscedastic Gaussian process regression. *Renewable Energy*, 148:1124–1136, 2020.

[106] Jack M. Wang, David J. Fleet, and Aaron Hertzmann. Gaussian process dynamical models for human motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):283–298, 2008.

[107] Shrihari Vasudevan, Fabio Ramos, Eric Nettleton, and Hugh Durrant-Whyte. Gaussian process modeling of large-scale terrain. *Journal of Field Robotics*, 26(10):812–840, 2009.

[108] Daniel Foreman-Mackey, Eric Agol, Sivaram Ambikasaran, and Ruth Angus. Fast and scalable Gaussian process modeling with applications to astronomical time series. *The Astronomical Journal*, 154(6):220, 2017.

[109] Oludare Isaac Abiodun, Aman Jantan, Abiodun Esther Omolara, Kemi Victoria Dada, Abubakar Malah Umar, Okafor Uchenwa Linus, Humaira Arshad, Abdullahi Aminu Kazaure, Usman Gana, and Muhammad Ubale Kiru. Comprehensive review of artificial neural network applications to pattern recognition. *IEEE Access*, 7:158820–158846, 2019.

[110] Abhishek Rawat, RN Yadav, and SC Shrivastava. Neural network applications in smart antenna arrays: A review. *AEU-International Journal of Electronics and Communications*, 66(11):903–912, 2012.

[111] Michael Egmont-Petersen, Dick de Ridder, and Heinz Handels. Image processing with neural networks—A review. *Pattern recognition*, 35(10):2279–2301, 2002.

[112] Xiaofang Hu, Gang Feng, Shukai Duan, and Lu Liu. A memristive multilayer cellular neural network with applications to image processing. *IEEE transactions on neural networks and learning systems*, 28(8):1889–1901, 2016.

[113] Daniël M Pelt and James A Sethian. A mixed-scale dense convolutional neural network for image analysis. *Proceedings of the National Academy of Sciences*, 115(2):254–259, 2018.

[114] José Naranjo-Torres, Marco Mora, Ruber Hernández-García, Ricardo J Barrientos, Claudio Fredes, and Andres Valenzuela. A review of convolutional neural network applied to fruit image processing. *Applied Sciences*, 10(10):3443, 2020.

[115] Naresh Singh, TN Singh, Avyaktanand Tiwary, and Kripa M Sarkar. Textural identification of basaltic rock mass using image processing and neural network. *Computational Geosciences*, 14(2):301–310, 2010.

[116] Yoav Goldberg. A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research*, 57:345–420, 2016.

[117] Yoav Goldberg. Neural network methods for natural language processing. *Synthesis lectures on human language technologies*, 10(1):1–309, 2017.

[118] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167, 2008.

[119] Ankit Kumar, Ozan Irsoy, Peter Ondruska, Mohit Iyyer, James Bradbury, Ishaan Gulrajani, Victor Zhong, Romain Paulus, and Richard Socher. Ask me anything: Dynamic memory networks for natural language processing. In *International conference on machine learning*, pages 1378–1387. PMLR, 2016.

[120] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In *Advances in neural information processing systems*, pages 6571–6583, 2018.

[121] Hamid Rahmanifard and Tatyana Plaksina. Application of artificial intelligence techniques in the petroleum industry: A review. *Artificial Intelligence Review*, 52(4):2295–2318, 2019.

[122] Oludare Isaac Abiodun, Aman Jantan, Abiodun Esther Omolara, Kemi Victoria Dada, Nachaat AbdElatif Mohamed, and Humaira Arshad. State-of-the-art in artificial neural network applications: A survey. *Heliyon*, 4(11):e00938, 2018.

[123] Bernard O Koopman. Hamiltonian systems and transformation in Hilbert space. *Proceedings of the national academy of sciences of the united states of america*, 17(5):315, 1931.

[124] Igor Mezić and Andrzej Banaszuk. Comparison of systems with complex behavior. *Physica D: Nonlinear Phenomena*, 197(1-2):101–133, 2004.

[125] Igor Mezić. Spectral properties of dynamical systems, model reduction and decompositions. *Nonlinear Dynamics*, 41(1):309–325, 2005.

[126] J. N. Kutz, S. L. Brunton, B. W. Brunton, and J. L. Proctor. *Dynamic Mode Decomposition: Data-Driven Modeling of Complex Systems*. SIAM, 2016.

[127] Jonathan H Tu. *Dynamic mode decomposition: Theory and applications*. PhD thesis, Princeton University, 2013.

[128] P. J. Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics*, 656:5–28, August 2010.

[129] Steven L Brunton, Bingni W Brunton, Joshua L Proctor, and J Nathan Kutz. Koopman

invariant subspaces and finite linear representations of nonlinear dynamical systems for control. *PloS one*, 11(2):e0150171, 2016.

[130] Stefan Klus, Péter Koltai, and Christof Schütte. On the numerical approximation of the Perron-Frobenius and Koopman operator. *arXiv preprint arXiv:1512.05997*, 2015.

[131] Matthew O Williams, Clarence W Rowley, and Ioannis G Kevrekidis. A kernel-based approach to data-driven Koopman spectral analysis. *arXiv preprint arXiv:1411.2260*, 2014.

[132] Patrick Héas, Cédric Herzet, and Benoit Combes. Generalized kernel-based dynamic mode decomposition. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3877–3881. IEEE, 2020.

[133] Frank Noé and Feliks Nuske. A variational approach to modeling slow processes in stochastic dynamical systems. *Multiscale Modeling & Simulation*, 11(2):635–655, 2013.

[134] Feliks Nüske, Reinhold Schneider, Francesca Vitalini, and Frank Noé. Variational tensor approach for approximating the rare-event kinetics of macromolecular systems. *The Journal of chemical physics*, 144(5):054105, 2016.

[135] Qianxiao Li, Felix Dietrich, Erik M Bollt, and Ioannis G Kevrekidis. Extended dynamic mode decomposition with dictionary learning: A data-driven adaptive spectral decomposition of the Koopman operator. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 27(10):103111, 2017.

[136] Bethany Lusch, J Nathan Kutz, and Steven L Brunton. Deep learning for universal linear embeddings of nonlinear dynamics. *Nature communications*, 9(1):1–10, 2018.

[137] Enoch Yeung, Soumya Kundu, and Nathan Hodas. Learning deep neural network representations for Koopman operators of nonlinear dynamical systems. In *2019 American Control Conference (ACC)*, pages 4832–4839. IEEE, 2019.

[138] Naoya Takeishi, Yoshinobu Kawahara, and Takehisa Yairi. Learning Koopman invariant subspaces for dynamic mode decomposition. *Advances in Neural Information Processing Systems*, 30, 2017.

[139] Yiqiang Han, Wenjian Hao, and Umesh Vaidya. Deep learning of Koopman representation for control. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pages 1890–1895. IEEE, 2020.

[140] Robert Julian Rabben, Sourav Ray, and Marcus Weber. Isokann: Invariant subspaces of Koopman operators learned by a neural network. *The Journal of Chemical Physics*, 153(11):114109, 2020.

[141] Omri Azencot, N Benjamin Erichson, Vanessa Lin, and Michael Mahoney. Forecasting sequential data using consistent Koopman autoencoders. In *International Conference on Machine Learning*, pages 475–485. PMLR, 2020.

[142] Alexandre Mauroy and Jorge Goncalves. Linear identification of nonlinear systems: A lifting technique based on the Koopman operator. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 6500–6505. IEEE, 2016.

[143] Matthew O Williams, Maziar S Hemati, Scott TM Dawson, Ioannis G Kevrekidis, and Clarence W Rowley. Extending data-driven Koopman analysis to actuated systems. *IFAC-PapersOnLine*, 49(18):704–709, 2016.

[144] Amit Surana. Koopman operator based observer synthesis for control-affine nonlinear systems. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 6492–6499. IEEE, 2016.

[145] Amit Surana and Andrzej Banaszuk. Linear observer synthesis for nonlinear systems using Koopman operator framework. *IFAC-PapersOnLine*, 49(18):716–723, 2016.

[146] Milan Korda and Igor Mezić. Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control. *Automatica*, 93:149–160, 2018.

[147] Ian Abraham, Gerardo De La Torre, and Todd D Murphey. Model-based control using Koopman operators. *arXiv preprint arXiv:1709.01568*, 2017.

[148] Giorgos Mamakoukas, Maria Castano, Xiaobo Tan, and Todd Murphey. Local Koopman operators for data-driven control of robotic systems. In *Robotics: Science and Systems*, 2019.

[149] Ian Abraham and Todd D Murphey. Active learning of dynamics for data-driven control using Koopman operators. *IEEE Transactions on Robotics*, 35(5):1071–1083, 2019.

[150] Daniel Bruder, Brent Gillespie, C David Remy, and Ram Vasudevan. Modeling and control of soft robots using the Koopman operator and model predictive control. *arXiv preprint arXiv:1902.02827*, 2019.

[151] Eurika Kaiser, J Nathan Kutz, and Steven L Brunton. Data-driven approximations of dynamical systems operators for control. In *The Koopman Operator in Systems and Control*, pages 197–234. Springer, 2020.

[152] Eurika Kaiser, J Nathan Kutz, and Steven L Brunton. Data-driven discovery of Koopman eigenfunctions for control. *Machine Learning: Science and Technology*, 2(3):035023, 2021.

[153] Joshua L Proctor, Steven L Brunton, and J Nathan Kutz. Generalizing Koopman theory to allow for inputs and control. *SIAM Journal on Applied Dynamical Systems*, 17(1):909–930, 2018.

[154] Kunihiko Taira, Steven L Brunton, Scott TM Dawson, Clarence W Rowley, Tim Colonius, Beverley J McKeon, Oliver T Schmidt, Stanislav Gordeyev, Vassilios Theofilis, and Lawrence S Ukeiley. Modal analysis of fluid flows: An overview. *Aiaa Journal*, 55(12):4013–4041, 2017.

[155] Kunihiko Taira, Maziar S Hemati, Steven L Brunton, Yiyang Sun, Karthik Duraisamy, Shervin Bagheri, Scott TM Dawson, and Chi-An Yeh. Modal analysis of fluid flows: Applications and outlook. *AIAA journal*, 58(3):998–1022, 2020.

[156] Samuel E Otto and Clarence W Rowley. Koopman operators for estimation and control of dynamical systems. *Annual Review of Control, Robotics, and Autonomous Systems*, 4:59–87, 2021.

[157] Steven L Brunton, Marko Budišić, Eurika Kaiser, and J Nathan Kutz. Modern Koopman theory for dynamical systems. *arXiv preprint arXiv:2102.12086*, 2021.

[158] Jer-Nan Juang and Richard S. Pappa. An eigensystem realization algorithm for modal parameter identification and model reduction. *Journal of Guidance, Control, and Dynamics*, 8(5):620–627, 1985.

[159] Zhanhua Ma, Sunil Ahuja, and Clarence W Rowley. Reduced-order models for control of fluids using the eigensystem realization algorithm. *Theoretical and Computational Fluid Dynamics*, 25(1):233–247, 2011.

[160] J.-N. Juang and H. Suzuki. An eigensystem realization algorithm in frequency domain for modal parameter identification. *Journal of Vibration, Acoustics, Stress, and Reliability in Design*, 110(1):24–29, 01 1988.

[161] Anas Almunif, Lingling Fan, and Zhixin Miao. A tutorial on data-driven eigenvalue identification: Prony analysis, matrix pencil, and eigensystem realization algorithm. *International Transactions on Electrical Energy Systems*, 30(4):e12283, 2020.

[162] Travis Askham and J Nathan Kutz. Variable projection methods for an optimized dynamic mode decomposition. *SIAM Journal on Applied Dynamical Systems*, 17(1):380–416, 2018.

[163] Naoya Takeishi, Yoshinobu Kawahara, Yasuo Tabei, and Takehisa Yairi. Bayesian dynamic mode decomposition. In *IJCAI*, pages 2814–2821, 2017.

[164] Joshua L Proctor, Steven L Brunton, and J Nathan Kutz. Dynamic mode decomposition with control. *SIAM Journal on Applied Dynamical Systems*, 15(1):142–161, 2016.

[165] Yoshinori Mizuno, Daniel Duke, Callum Atkinson, and Julio Soria. Investigation of wall-bounded turbulent flow using dynamic mode decomposition. *Journal of Physics: Conference Series*, 318:042040, 12 2011.

[166] Joshua L Proctor and Philip A Eckhoff. Discovering dynamic patterns from infectious disease data using dynamic mode decomposition. *International health*, 7(2):139–145, 2015.

[167] Roy Taylor, J Nathan Kutz, Kyle Morgan, and Brian A Nelson. Dynamic mode decomposition for plasma diagnostics and validation. *Review of Scientific Instruments*, 89(5):053501, 2018.

[168] Alan A Kaptanoglu, Kyle D Morgan, Chris J Hansen, and Steven L Brunton. Characterizing magnetized plasmas with dynamic mode decomposition. *Physics of Plasmas*, 27(3):032108, 2020.

[169] Saeed Asgari, Xiao Hu, Michael Tsuk, and Shailendra Kaushik. Application of POD plus LTI ROM to battery thermal modeling: SISO case. *SAE International Journal of Commercial Vehicles*, 7(2014-01-1843):278–285, 2014.

[170] Hugo FS Lui and William R Wolf. Construction of reduced-order models for fluid flows using deep feedforward neural networks. *Journal of Fluid Mechanics*, 872:963–994, 2019.

[171] Clarence W Rowley, Tim Colonius, and Richard M Murray. Model reduction for compressible flows using POD and Galerkin projection. *Physica D: Nonlinear Phenomena*, 189(1-2):115–129, 2004.

[172] Julien Weiss. A tutorial on the proper orthogonal decomposition. In *AIAA Aviation 2019 Forum*, page 3333, 2019.

[173] Alan A Kaptanoglu, Kyle D Morgan, Chris J Hansen, and Steven L Brunton. Physics-constrained, low-dimensional models for magnetohydrodynamics: First-principles and data-driven approaches. *Physical Review E*, 104(1):015206, 2021.

[174] Gal Berkooz, Philip Holmes, and John L Lumley. The proper orthogonal decomposition in the analysis of turbulent flows. *Annual review of fluid mechanics*, 25(1):539–575, 1993.

[175] Mehmet Onder Efe and Hitay Ozbay. Proper orthogonal decomposition for reduced order modeling: 2D heat flow. In *Proceedings of 2003 IEEE Conference on Control Applications, 2003. CCA 2003.*, volume 2, pages 1273–1277. IEEE, 2003.

[176] Stefania Fresca and Andrea Manzoni. POD-DL-ROM: Enhancing deep learning-based reduced order models for nonlinear parametrized PDEs by proper orthogonal decomposition. *Computer Methods in Applied Mechanics and Engineering*, 388:114181, 2022.

[177] Kadierdan Kaheman, J Nathan Kutz, and Steven L Brunton. SINDy-PI: A robust algorithm for parallel implicit sparse identification of nonlinear dynamics. *Proceedings of the Royal Society A*, 476(2242):20200279, 2020.

[178] Kadierdan Kaheman, Eurika Kaiser, Benjamin Strom, J Nathan Kutz, and Steven L Brunton. Learning discrepancy models from experimental data. *Conference on Decision and Control*, 2019.

[179] Kadierdan Kaheman, Urban Fasel, Jason J Bramburger, Benjamin Strom, J Nathan Kutz, and Steven L Brunton. The experimental multi-arm pendulum on a cart: A benchmark system for chaos, learning, and control. *arXiv preprint arXiv:2205.06231*, 2022.

[180] Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the national academy of sciences*, 113(15):3932–3937, 2016.

[181] Niall M Mangan, Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Inferring biological networks by sparse identification of nonlinear dynamics. *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, 2(1):52–63, 2016.

[182] Qing Qu, Ju Sun, and John Wright. Finding a sparse vector in a subspace: Linear sparsity using alternating directions. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, pages 3401–3409, Cambridge, MA, USA, 2014. MIT Press.

[183] M. Gavish and D. L. Donoho. The optimal hard threshold for singular values is $4/\sqrt{3}$. *IEEE Transactions on Information Theory*, 60(8):5040–5053, 2014.

[184] Linan Zhang and Hayden Schaeffer. On the convergence of the SINDy algorithm. *Multiscale Modeling & Simulation*, 17(3):948–972, 2019.

[185] Jonathan H Horrocks. Sparse identification of epidemiological models from empirical data. Master's thesis, University of Waterloo, 2018.

[186] Weijie Su, Małgorzata Bogdan, Emmanuel Candes, et al. False discoveries occur early on the LASSO path. *The Annals of statistics*, 45(5):2133–2150, 2017.

[187] Robert Tibshirani. Regression shrinkage and selection via the LASSO. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

[188] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[189] Hui Zou. The adaptive LASSO and its oracle properties. *Journal of the American statistical association*, 101(476):1418–1429, 2006.

[190] Alexandre Cortiella, Kwang-Chun Park, and Alireza Doostan. Sparse identification of nonlinear dynamical systems via reweighted ℓ_1-regularized least squares. *Computer Methods in Applied Mechanics and Engineering*, 376:113620, 2021.

[191] W. Pan, Y. Yuan, J. Gonçalves, and G. Stan. A sparse Bayesian approach to the identification of nonlinear state-space systems. *IEEE Transactions on Automatic Control*, 61(1):182–187, January 2016.

[192] Scott Shaobing Chen, David L Donoho, and Michael A Saunders. Atomic decomposition by basis pursuit. *SIAM review*, 43(1):129–159, 2001.

[193] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the royal statistical society: series B (statistical methodology)*, 67(2):301–320, 2005.

[194] Moritz Hoffmann, Christoph Fröhner, and Frank Noé. Reactive SINDy: Discovering governing reactions from concentration data. *The Journal of chemical physics*, 150(2):025101, 2019.

[195] Markus Quade, Markus Abel, J Nathan Kutz, and Steven L Brunton. Sparse identification of nonlinear dynamics for rapid model recovery. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 28(6):063116, 2018.

[196] Kai Fukami, Takaaki Murata, Kai Zhang, and Koji Fukagata. Sparse identification of nonlinear dynamics with low-dimensionalized flow representations. *Journal of Fluid Mechanics*, 926, 2021.

[197] Arthur E Hoerl and Robert W Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

[198] Urban Fasel, J Nathan Kutz, Bingni W Brunton, and Steven L Brunton. Ensemble-SINDy: Robust sparse model discovery in the low-data, high-noise limit, with active learning and control. *arXiv preprint arXiv:2111.10992*, 2021.

[199] Yuqiang Wu. Error processing of sparse identification of nonlinear dynamical systems via l_∞ approximation. *arXiv preprint arXiv:2107.06142*, 2021.

[200] Lorenzo Boninsegna, Feliks Nüske, and Cecilia Clementi. Sparse learning of stochastic dynamical equations. *The Journal of Chemical Physics*, 148(24):241723, 2018.

[201] E Paulo Alves and Frederico Fiuza. Data-driven discovery of reduced plasma physics models from fully-kinetic simulations. *arXiv preprint arXiv:2011.01927*, 2020.

[202] Chaitanya Joshi, Sattvic Ray, Linnea Lemma, Minu Varghese, Graham Sharp, Zvonimir Dogic, Aparna Baskaran, and Michael F Hagan. Data-driven discovery of active nematic hydrodynamics. *arXiv preprint arXiv:2202.12854*, 2022.

[203] Y.C. Pati, R. Rezaiifar, and P.S. Krishnaprasad. Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In *Proceedings of 27th Asilomar Conference on Signals, Systems and Computers*, pages 40–44 vol.1, 1993.

[204] Yannis Pantazis and Ioannis Tsamardinos. A unified approach for sparse dynamical system inference from temporal measurements. *Bioinformatics*, 35(18):3387–3396, 2019.

[205] Yuzhu Guo, LZ Guo, Stephen A Billings, and Hua-Liang Wei. An iterative orthogonal forward regression algorithm. *International Journal of Systems Science*, 46(5):776–789, 2015.

[206] Christophe Couvreur and Yoram Bresler. On the optimality of the backward greedy algorithm for the subset selection problem. *SIAM Journal on Matrix Analysis and Applications*, 21(3):797–808, 2000.

[207] Tong Zhang. Adaptive forward-backward greedy algorithm for learning sparse representations. *IEEE transactions on information theory*, 57(7):4689–4708, 2011.

[208] Tong Zhang. Adaptive forward-backward greedy algorithm for sparse learning with linear models. *Advances in neural information processing systems*, 21, 2008.

[209] Brian M de Silva, David M Higdon, Steven L Brunton, and J Nathan Kutz. Discovery of physics from data: Universal laws and discrepancies. *Frontiers in artificial intelligence*, 3:25, 2020.

[210] Hayden Schaeffer, Giang Tran, and Rachel Ward. Learning dynamical systems and bifurcation via group sparsity. *arXiv preprint arXiv:1709.01558*, 2017.

[211] S. Rudy, A. Alla, S. L. Brunton, and J. N. Kutz. Data-driven identification of parametric partial differential equations. *SIAM Journal on Applied Dynamical Systems*, 18(2):643–660, 2019.

[212] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. A note on the group LASSO and a sparse group LASSO. *arXiv preprint arXiv:1001.0736*, 2010.

[213] Volker Roth and Bernd Fischer. The group-LASSO for generalized linear models: Uniqueness of solutions and efficient algorithms. In *Proceedings of the 25th international conference on Machine learning*, pages 848–855, 2008.

[214] Peng Zheng, Travis Askham, Steven L Brunton, J Nathan Kutz, and Aleksandr Y Aravkin. Sparse relaxed regularized regression: SR3. *IEEE Access*, 7(1):1404–1423, 2019.

[215] Kathleen Champion, Peng Zheng, Aleksandr Y Aravkin, Steven L Brunton, and J Nathan Kutz. A unified sparse optimization framework to learn parsimonious physics-informed models from data. *IEEE Access*, 8:169259–169271, 2020.

[216] Peng Zheng, Travis Askham, Steven L Brunton, J Nathan Kutz, and Aleksandr Y Aravkin. A unified framework for sparse relaxed regularized regression: SR3. *IEEE Access*, 7:1404–1423, 2018.

[217] Yisha Lu, Wei Xu, Yiyu Jiao, and Minjuan Yuan. Sparse identification of nonlinear dynamical systems via non-convex penalty least squares. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 32(2):023113, 2022.

[218] R. Fuentes, N. Dervilis, K. Worden, and E.J. Cross. Efficient parameter identification and model selection in nonlinear dynamical systems via sparse Bayesian learning. *Journal of Physics: Conference Series*, 1264(1):012050, jul 2019.

[219] Dimitris Bertsimas, Jean Pauphilet, and Bart Van Parys. Sparse regression: Scalable algorithms and empirical performance. *Statistical Science*, 35(4):555–578, 2020.

[220] Alexandre Cortiella, Kwang-Chun Park, and Alireza Doostan. A priori denoising strategies for sparse identification of nonlinear dynamical systems: A comparative study. *arXiv preprint arXiv:2201.12683*, 2022.

[221] Abd AlRahman R AlMomani, Jie Sun, and Erik Bollt. How entropic regression beats the outliers problem in nonlinear system identification. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 30(1):013107, 2020.

[222] N. M. Mangan, J. N. Kutz, S. L. Brunton, and J. L. Proctor. Model selection for dynamical systems via sparse regression and information criteria. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Science*, 473(2204):20170009, Aug 2017.

[223] Hirotugu Akaike. A new look at the statistical model identification. *IEEE transactions on automatic control*, 19(6):716–723, 1974.

[224] Hirotogu Akaike. Information theory and an extension of the maximum likelihood principle. In *Selected papers of hirotugu akaike*, pages 199–213. Springer, 1998.

[225] Gideon Schwarz. Estimating the dimension of a model. *Ann. Statist.*, 6(2):461–464, 03 1978.

[226] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[227] Brian M de Silva, Kathleen Champion, Markus Quade, Jean-Christophe Loiseau, J Nathan Kutz, and Steven L Brunton. PySINDy: A Python package for the sparse identification of nonlinear dynamics from data. *arXiv preprint arXiv:2004.08424*, 2020.

[228] Alan A Kaptanoglu, Brian M de Silva, Urban Fasel, Kadierdan Kaheman, Jared L Callaham, Charles B Delahunt, Kathleen Champion, Jean-Christophe Loiseau, J Nathan Kutz, and Steven L Brunton. PySINDy: A comprehensive Python package for robust sparse system identification. *arXiv preprint arXiv:2111.08481*, 2021.

[229] Sheng Zhang and Guang Lin. SubTSBR to tackle high noise and outliers for data-driven discovery of differential equations. *Journal of Computational Physics*, 428:109962, 2021.

[230] Fahim Abdullah, Zhe Wu, and Panagiotis D Christofides. Handling noisy data in sparse model identification using subsampling and co-teaching. *Computers & Chemical Engineering*, 157:107628, 2022.

[231] Charles B Delahunt and J Nathan Kutz. A toolkit for data-driven discovery of governing equations in high-noise regimes. *arXiv preprint arXiv:2111.04870*, 2021.

[232] Daniel R. Gurevich, Patrick A. K. Reinbold, and Roman O. Grigoriev. Robust and optimal sparse regression for nonlinear PDE models. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 29(10):103113, 2019.

[233] Daniel A Messenger and David M Bortz. Weak SINDy: Galerkin-based data-driven model selection. *arXiv preprint arXiv:2005.04339*, 2020.

[234] Daniel A Messenger and David M Bortz. Weak SINDy for partial differential equations. *Journal of Computational Physics*, 443:110525, 2021.

[235] Daniel A Messenger, Emiliano Dall'Anese, and David M Bortz. Online weak-form sparse identification of partial differential equations. *arXiv preprint arXiv:2203.03979*, 2022.

[236] Hayden Schaeffer and Scott G. McCalla. Sparse model selection via integral terms. *Phys. Rev. E*, 96:023302, Aug 2017.

[237] Damien Guého, Puneet Singla, and Robert G Melton. Data-driven sparse approximation for the identification of nonlinear dynamical systems: Application in astrodynamics. *Spaceflight Mechanics 2020*, 2020.

[238] Miaomiao Lin, Changming Cheng, Zhike Peng, Xingjian Dong, Yegao Qu, and Guang Meng. Nonlinear dynamical system identification using the sparse regression and separable least squares methods. *Journal of Sound and Vibration*, 505:116141, 2021.

[239] Pawan Goyal and Peter Benner. Discovery of nonlinear dynamical systems using a Runge-Kutta inspired dictionary-based sparse regression approach. *arXiv preprint arXiv:2105.04869*, 2021.

[240] Kookjin Lee, Nathaniel Trask, and Panos Stinis. Structure-preserving sparse identification of nonlinear dynamics for data-driven modeling. *arXiv preprint arXiv:2109.05364*, 2021.

[241] Sheng Zhang and Guang Lin. Robust data-driven discovery of governing physical laws with error bars. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 474(2217):20180305, 2018.

[242] Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

[243] Peter Lukas Bühlmann. Bagging, subagging and bragging for improving some prediction algorithms. In *Research report/Seminar für Statistik, Eidgenössische Technische Hochschule (ETH)*, volume 113. Seminar für Statistik, Eidgenössische Technische Hochschule (ETH), Zürich, 2003.

[244] Peter Bühlmann. Bagging, boosting and ensemble methods. In *Handbook of computational statistics*, pages 985–1022. Springer, 2012.

[245] Anders Krogh and Jesper Vedelsby. Neural network ensembles, cross validation, and active learning. *Advances in neural information processing systems*, 7, 1994.

[246] Xingquan Zhu, Peng Zhang, Xiaodong Lin, and Yong Shi. Active learning from stream data using optimal weight classifier ensemble. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 40(6):1607–1621, 2010.

[247] Burr Settles. Active learning literature survey. 07 2010.

[248] Burr Settles. From theories to queries: Active learning in practice. In *Active learning and experimental design workshop in conjunction with AISTATS 2010*, pages 1–18. JMLR Workshop and Conference Proceedings, 2011.

[249] Zhe Wu, Junwei Luo, David Rincon, and Panagiotis D Christofides. Machine learning-based predictive control using noisy data: Evaluating performance and robustness via a large-scale process simulator. *Chemical Engineering Research and Design*, 168:275–287, 2021.

[250] Zhe Wu, David Rincon, Junwei Luo, and Panagiotis D Christofides. Machine learning modeling and predictive control of nonlinear processes using noisy data. *AIChE Journal*, 67(4):e17164, 2021.

[251] Giang Tran and Rachel Ward. Exact recovery of chaotic systems from highly corrupted data. *Multiscale Modeling & Simulation*, 15(3):1108–1129, 2017.

[252] Gustavo T Naozuka, Heber L Rocha, Renato S Silva, and Regina C Almeida. SINDy-SA: Enhancing nonlinear system identification with sensitivity analysis. *Research Gate*, 2022.

[253] Alejandro Carderera, Sebastian Pokutta, Christof Schütte, and Martin Weiser. CINDy: Conditional gradient-based identification of non-linear dynamics–noise-robust recovery. *arXiv preprint arXiv:2101.02630*, 2021.

[254] Jean-Christophe Loiseau and Steven L Brunton. Constrained sparse Galerkin regression. *Journal of Fluid Mechanics*, 838:42–67, 2018.

[255] Alan A Kaptanoglu, Jared L Callaham, Aleksandr Aravkin, Christopher J Hansen, and Steven L Brunton. Promoting global stability in data-driven models of quadratic nonlinear dynamics. *Physical Review Fluids*, 6(9):094401, 2021.

[256] Seth M Hirsh, David A Barajas-Solano, and J Nathan Kutz. Sparsifying priors for Bayesian uncertainty quantification in model discovery. *arXiv preprint arXiv:2107.02107*, 2021.

[257] Yu-Xin Jiang, Xiong Xiong, Shuo Zhang, Jia-Xiang Wang, Jia-Chun Li, and Lin Du. Modeling and prediction of the transmission dynamics of COVID-19 based on the SINDy-LM method. *Nonlinear Dynamics*, 105(3):2775–2794, 2021.

[258] Manolis IA Lourakis et al. A brief description of the Levenberg-Marquardt algorithm implemented by levmar. *Foundation of Research and Technology*, 4(1):1–6, 2005.

[259] RK Al Seyab and Yi Cao. Nonlinear system identification for predictive control using continuous time recurrent neural networks and automatic differentiation. *Journal of Process Control*, 18(6):568–581, 2008.

[260] Andrea Saltelli, Marco Ratto, Terry Andres, Francesca Campolongo, Jessica Cariboni, Debora Gatelli, Michaela Saisana, and Stefano Tarantola. *Global sensitivity analysis: the primer*. John Wiley & Sons, 2008.

[261] Thaynã França, Arthur Martins Barbosa Braga, and Helon Vicente Hultmann Ayala. Feature engineering to cope with noisy data in sparse identification. *Expert Systems with Applications*, 188:115995, 2022.

[262] Hemant Ishwaran and J Sunil Rao. Spike and slab variable selection: Frequentist and Bayesian strategies. *The Annals of Statistics*, 33(2):730–773, 2005.

[263] David Madigan and Adrian E Raftery. Model selection and accounting for model uncertainty in graphical models using Occam's window. *Journal of the American Statistical Association*, 89(428):1535–1546, 1994.

[264] Toby J Mitchell and John J Beauchamp. Bayesian variable selection in linear regression. *Journal of the american statistical association*, 83(404):1023–1032, 1988.

[265] Juho Piironen and Aki Vehtari. Sparsity information and regularization in the horseshoe and other shrinkage priors. *Electronic Journal of Statistics*, 11(2):5018–5051, 2017.

[266] Shanwu Li, Eurika Kaiser, Shujin Laima, Hui Li, Steven L Brunton, and J Nathan Kutz. Discovering time-varying aerodynamics of a prototype bridge by sparse identification of nonlinear dynamical systems. *Physical Review E*, 100(2):022220, 2019.

[267] Niall M Mangan, Travis Askham, Steven L Brunton, J Nathan Kutz, and Joshua L Proctor. Model selection for hybrid dynamical systems via sparse regression. *Proceedings of the Royal Society A*, 475(2223):20180534, 2019.

[268] Gerhard Schreck, Gregor Thiele, Arne Fey, and Jörg Krüger. Robust system identification for hysteresis-controlled devices using SINDy. In *IECON 2020 The 46th Annual Conference of the IEEE Industrial Electronics Society*, pages 4820–4824. IEEE, 2020.

[269] Jason J Bramburger, Daniel Dylewsky, and J Nathan Kutz. Sparse identification of slow timescale dynamics. *Physical Review E*, 102(2):022204, 2020.

[270] Takaaki Murata, Kai Fukami, and Koji Fukagata. CNN-SINDy based reduced order modeling of unsteady flow fields. In *Fluids Engineering Division Summer Meeting*, volume 59032, page V002T02A074. American Society of Mechanical Engineers, 2019.

[271] Megan R Ebers, Katherine M Steele, and J Nathan Kutz. Discrepancy modeling framework: Learning missing physics, modeling systematic residuals, and disambiguating between deterministic and random effects. *arXiv preprint arXiv:2203.05164*, 2022.

[272] Markus Quade. *Symbolic regression for identification, prediction, and control of dynamical systems*. PhD thesis, Universität Potsdam, 2018.

[273] Saeideh Khatiry Goharoodi, Kevin Dekemele, Mia Loccufier, Luc Dupre, and Guillaume Crevecoeur. Evolutionary-based sparse regression for the experimental identification of Duffing oscillator. *Mathematical Problems in Engineering*, 2020, 2020.

[274] Joseph Bakarji, Kathleen Champion, J Nathan Kutz, and Steven L Brunton. Discovering governing equations from partial measurements with deep delay autoencoders. *arXiv preprint arXiv:2201.05136*, 2022.

[275] RV Sanjika Devi and Dhanesh G Kurup. Sparse identification of memory effects and nonlinear dynamics for developing parsimonious behavioral models of RF power amplifiers. In *2017 IEEE MTT-S International Microwave and RF Conference (IMaRC)*, pages 1–4. IEEE, 2017.

[276] Zhilu Lai, Ignacio Alzugaray, Margarita Chli, and Eleni Chatzi. Full-field structural monitoring using event cameras and physics-informed sparse identification. *Mechanical Systems and Signal Processing*, 145:106905, 2020.

[277] Mariia Sorokina, Stylianos Sygletos, and Sergei Turitsyn. Sparse identification for nonlinear optical communication systems: SINO method. *Optics express*, 24(26):30433–30443, 2016.

[278] FEI TAO, XIN LIU, HAODONG DU, and WENBIN YU. Discovering failure criteria of composites by sparse identification and compressed sensing. In *Proceedings of the American Society for Composites—Thirty-Sixth Technical Conference on Composite Materials*, 2021.

[279] Gregor Thiele, Arne Fey, David Sommer, and Jörg Krüger. System identification of a hysteresis-controlled pump system using SINDy. In *2020 24th International Conference on System Theory, Control and Computing (ICSTCC)*, pages 457–464. IEEE, 2020.

[280] Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Sparse identification of nonlinear dynamics with control (SINDyc). *IFAC-PapersOnLine*, 49(18):710–715, 2016.

[281] Eurika Kaiser, J Nathan Kutz, and Steven L Brunton. Sparse identification of nonlinear dynamics for model predictive control in the low-data limit. *Proceedings of the Royal Society of London A*, 474(2219), 2018.

[282] Hasan Matpan. Data driven model discovery and control of longitudinal missile dynamics. Master's thesis, Middle East Technical University, 2021.

[283] Fahim Abdullah, Zhe Wu, and Panagiotis D Christofides. Sparse-identification-based model predictive control of nonlinear two-time-scale processes. *Computers & Chemical Engineering*, 153:107411, 2021.

[284] S. Emad Askarinejad, Ali Reza Fahim, Mohammad Reza Hairi Yazdi, and Mehdi Tale Masouleh. Data-driven identification of the Jacobian matrix of a 2- DoF spherical parallel manipulator. *2019 7th International Conference on Robotics and Mechatronics (ICRoM)*, pages 229–234, 2019.

[285] Daniel E Shea, Steven L Brunton, and J Nathan Kutz. SINDy-BVP: Sparse identification of nonlinear dynamics for boundary value problems. *Physical Review Research*, 3(2):023255, 2021.

[286] Ghazaale Leylaz, Shuo Wang, and Jian-Qiao Sun. Identification of nonlinear dynamical systems with time delay. *International Journal of Dynamics and Control*, 10(1):13–24, 2022.

[287] Stephan Thaler, Ludger Paehler, and Nikolaus A Adams. Sparse identification of truncation errors. *Journal of Computational Physics*, 397:108851, 2019.

[288] Jason J Bramburger and J Nathan Kutz. Poincaré maps for multiscale physics discovery and nonlinear Floquet theory. *Physica D: Nonlinear Phenomena*, 408:132479, 2020.

[289] S Khatiry Goharoodi, Kevin Dekemele, Luc Dupre, Mia Loccufier, and Guillaume Crevecoeur. Sparse identification of nonlinear Duffing oscillator from measurement data. *IFAC-PapersOnLine*, 51(33):162–167, 2018.

[290] Scott TM Dawson and Steven L Brunton. Improved approximations to Wagner function using sparse identification of nonlinear dynamics. *AIAA Journal*, pages 1–17, 2021.

[291] Chong Sun, Tian Tian, Xiaocheng Zhu, and Zhaohui Du. Sparse identification of nonlinear unsteady aerodynamics of the oscillating airfoil. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, 235(7):809–824, 2021.

[292] Daniel A Messenger and David M Bortz. Learning mean-field equations from particle data using WSINDy. *arXiv preprint arXiv:2110.07756*, 2021.

[293] Yifei Guan, Steven L Brunton, and Igor Novosselov. Sparse nonlinear models of chaotic electroconvection. *Royal Society Open Science*, 8(8):202367, 2021.

[294] Alan A Kaptanoglu, Kyle D Morgan, Christopher J Hansen, and Steven L Brunton. The structure of global conservation laws in Galerkin plasma models. *arXiv preprint arXiv:2101.03436*, 2021.

[295] Ariana Mendible, James Koch, Henning Lange, Steven L Brunton, and J Nathan Kutz. Data-driven modeling of rotating detonation waves. *Physical Review Fluids*, 6(5):050507, 2021.

[296] Tomoyuki Suzuki, Akira Kano, and Kenji Hirohata. Deriving thermal model from data by sparse identification based on physical laws. In *ASME International Mechanical Engineering Congress and Exposition*, volume 85673, page V011T11A003. American Society of Mechanical Engineers, 2021.

[297] Jean-Christophe Loiseau, Bernd R Noack, and Steven L Brunton. Sparse reduced-order modelling: Sensor-based dynamics to full-state estimation. *Journal of Fluid Mechanics*, 844:459–490, 2018.

[298] Patrick Gelß, Stefan Klus, Jens Eisert, and Christof Schütte. Multidimensional approximation of nonlinear dynamical systems. *Journal of Computational and Nonlinear Dynamics*, 14(6), 2019.

[299] Bhavana Bhadriraju, Mohammed Saad Faizan Bangi, Abhinav Narasingam, and Joseph Sang-Il Kwon. Operable adaptive sparse identification of systems: Application to chemical processes. *AIChE Journal*, 66(11):e16980, 2020.

[300] Magnus Dam, Morten Brøns, Jens Juul Rasmussen, Volker Naulin, and Jan S Hesthaven. Sparse identification of a predator-prey system from simulation data of a convection model. *Physics of Plasmas*, 24(2):022310, 2017.

[301] Harshavardhan Subramanian. Combining scientific computing and machine learning techniques to model longitudinal outcomes in clinical trials., 2021.

[302] Hayden Schaeffer. Learning partial differential equations via data discovery and sparse optimization. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 473(2197):20160446, 2017.

[303] Abhinav Narasingam and Joseph Sang-Il Kwon. Data-driven identification of interpretable reduced-order models using sparse regression. *Computers & Chemical Engineering*, 119:101–111, 2018.

[304] Zhilu Lai and Satish Nagarajaiah. Sparse structural system identification method for nonlinear dynamic systems with hysteresis/inelastic behavior. *Mechanical Systems and Signal Processing*, 117:813–842, 2019.

[305] Kailiang Wu and Dongbin Xiu. Numerical aspects for approximating governing equations using data. *Journal of Computational Physics*, 384:200–221, 2019.

[306] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 31(2):210–227, 2009.

[307] Quanmin Zhu, Yongji Wang, Dongya Zhao, Shaoyuan Li, and Stephen A Billings. Review of rational (total) nonlinear dynamic system modelling, identification, and control. *International Journal of Systems Science*, 46(12):2122–2133, 2015.

[308] Quanmin Zhu, Li Liu, Weicun Zhang, and Shaoyuan Li. Control of complex nonlinear dynamic rational systems. *Complexity*, 2018, 2018.

[309] Patrick AK Reinbold, Daniel R Gurevich, and Roman O Grigoriev. Using noisy or incomplete data to discover models of spatiotemporal dynamics. *Physical Review E*, 101(1):010203, 2020.

[310] Rick Chartrand. Numerical differentiation of noisy, nonsmooth data. *ISRN Applied Mathematics*, 2011, 2011.

[311] Michael Grant and Stephen Boyd. CVX: Matlab software for disciplined convex programming, version 2.1. `http://cvxr.com/cvx`, March 2014.

[312] Michael Grant and Stephen Boyd. Graph implementations for nonsmooth convex programs. In V. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pages 95–110. Springer-Verlag Limited, 2008. `http://stanford.edu/~boyd/graph_dcp.html`.

[313] Kenneth A. Johnson and Roger S. Goody. The original Michaelis constant: Translation of the 1913 Michaelis–Menten paper. *Biochemistry*, 50(39):8264–8269, 2011.

[314] Athel Cornish-Bowden. One hundred years of Michaelis–Menten kinetics. *Perspectives in Science*, 4:3–9, 2015.

[315] Michael D Schmidt, Ravishankar R Vallabhajosyula, Jerry W Jenkins, Jonathan E Hood, Abhishek S Soni, John P Wikswo, and Hod Lipson. Automated refinement and inference of analytical models for metabolic networks. *Physical biology*, 8(5):055011, 2011.

[316] Vladimir K Vanag. Waves and patterns in reaction–diffusion systems. Belousov–Zhabotinsky reaction in water-in-oil microemulsions. *Physics-Uspekhi*, 47(9):923, 2004.

[317] Lloyd N Trefethen. *Spectral Methods in MATLAB*, volume 10, chapter 8. Chebyshev Series and the FFT, pages 75–86. SIAM, 2000.

[318] Samuel H Rudy, J Nathan Kutz, and Steven L Brunton. Deep learning of dynamics and signal-noise decomposition with time-stepping constraints. *Journal of Computational Physics*, 396:483–506, 2019.

[319] Atılım Günes Baydin, Barak A Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. Automatic differentiation in machine learning: A survey. *The Journal of Machine Learning Research*, 18(1):5595–5637, 2017.

[320] Jean-Christophe Loiseau. Data-driven modeling of the chaotic thermal convection in an annular thermosyphon. *Theoretical and Computational Fluid Dynamics*, pages 1–27, 2020.

[321] Yosef El Sayed M, Richard Semaan, and Rolf Radespiel. Sparse modeling of the lift gains of a high-lift configuration with periodic Coanda blowing. In *2018 AIAA Aerospace Sciences Meeting*, page 1054, 2018.

[322] Nan Deng, Bernd R Noack, Marek Morzyński, and Luc R Pastur. Low-order model for successive bifurcations of the fluidic pinball. *Journal of Fluid Mechanics*, 884, 2020.

[323] Martin Schmelzer, Richard P Dwight, and Paola Cinnella. Discovery of algebraic Reynolds-stress models using sparse symbolic regression. *Flow, Turbulence and Combustion*, 104(2):579–603, 2020.

[324] Shaowu Pan, Nicholas Arnold-Medabalimi, and Karthik Duraisamy. Sparsity-promoting algorithms for the discovery of informative Koopman invariant subspaces. *arXiv preprint arXiv:2002.10637*, 2020.

[325] S Beetham and J Capecelatro. Formulating turbulence closures using sparse regression with embedded form invariance. *arXiv preprint arXiv:2003.12884*, 2020.

[326] Kathleen P Champion, Steven L Brunton, and J Nathan Kutz. Discovery of nonlinear multiscale systems: Sampling strategies and embeddings. *SIAM Journal on Applied Dynamical Systems*, 18(1):312–333, 2019.

[327] Hayden Schaeffer, Giang Tran, and Rachel Ward. Extracting sparse high-dimensional dynamics from limited data. *SIAM Journal on Applied Mathematics*, 78(6):3279–3295, 2018.

[328] A Goeßmann, M Götte, Ingo Roth, Ryan Sweke, Gitta Kutyniok, and Jens Eisert. Tensor network approaches for learning non-linear dynamical laws. *arXiv preprint arXiv:2002.12388*, 2020.

[329] Floris van Breugel, J. Nathan Kutz, and Bing Brunton. Numerical differentiation of noisy data: A unifying multi-objective optimization framework. *arXiv preprint arXiv:2009.01911*, 2020.

[330] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, pages 265–283, 2016.

[331] Christopher Rackauckas and Qing Nie. DifferentialEquations.jl–A performant and feature-rich ecosystem for solving differential equations in Julia. *Journal of Open Research Software*, 5(1), 2017.

[332] Bart Van Merriënboer, Olivier Breuleux, Arnaud Bergeron, and Pascal Lamblin. Automatic differentiation in ML: Where we are and where we should be going. In *Advances in neural information processing systems*, pages 8757–8767, 2018.

[333] Samuel H Rudy, Steven L Brunton, and J Nathan Kutz. Smoothing and parameter estimation by soft-adherence to governing equations. *Journal of Computational Physics*, 398:108860, 2019.

[334] Gert-Jan Both, Subham Choudhury, Pierre Sens, and Remy Kusters. DeepMoD: Deep learning for model discovery in noisy data. *Journal of Computational Physics*, 428:109985, 2021.

[335] Christopher Rackauckas, Yingbo Ma, Julius Martensen, Collin Warner, Kirill Zubov, Rohit Supekar, Dominic Skinner, and Ali Ramadhan. Universal differential equations for scientific machine learning. *arXiv preprint arXiv:2001.04385*, 2020.

[336] Henning Lange, Steven L Brunton, and Nathan Kutz. From Fourier to Koopman: Spectral methods for long-term time series prediction. *arXiv preprint arXiv:2004.00574*, 2020.

[337] Mike Innes. Flux: Elegant machine learning with Julia. *Journal of Open Source Software*, 2018.

[338] Samuel S Schoenholz, Ekin D Cubuk, and MD Jax. End-to-end differentiable, hardware accelerated, molecular dynamics in pure Python. *arXiv preprint arXiv:1912.04232*, 2018.

[339] Carl P Goodrich, Ella M King, Samuel S Schoenholz, Ekin D Cubuk, and Michael P Brenner. Designing self-assembling kinetics with differentiable statistical physics models. *Proceedings of the National Academy of Sciences*, 118(10), 2021.

[340] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[341] Angelo Vulpiani, Fabio Cecconi, and Massimo Cencini. *Chaos: From simple models to complex systems*, volume 17. World Scientific, 2009.

[342] Thomas Cokelaer, Brian, Caio Eadi Stringari, Eike Broda, and Elmar Pruesse. cokelaer/Fitter: v1.2.3 synchronised on pypi, August 2020.

[343] J. E. Marsden and T. S. Ratiu. *Introduction to mechanics and symmetry*. Springer-Verlag, 2nd edition, 1999.

[344] Eduardo F Camacho and Carlos Bordons Alba. *Model predictive control*. Springer Science & Business Media, 2013.

[345] Utku Eren, Anna Prach, Başaran Bahadır Koçer, Saša V Raković, Erdal Kayacan, and Behçet Açıkmeşe. Model predictive control in aerospace systems: Current state and opportunities. *Journal of Guidance, Control, and Dynamics*, 2017.

[346] Viviane Botelho, Jorge Otávio Trierweiler, Marcelo Farenzena, and Ricardo Duraiski. Methodology for detecting model–plant mismatches affecting model predictive

control performance. *Industrial & Engineering Chemistry Research*, 54(48):12072–12085, Nov 2015.

[347] Marc C Kennedy and Anthony O'Hagan. Bayesian calibration of computer models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(3):425–464, 2001.

[348] Paul D Arendt, Daniel W Apley, and Wei Chen. Quantification of model uncertainty: Calibration, model discrepancy, and identifiability. *Journal of Mechanical Design*, 134(10), 2012.

[349] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Journal of Fluids Engineering*, 82(1):35–45, 1960.

[350] B. Kim, D. Necsulescu *, and J. Sasiadek. Autonomous mobile robot model predictive control. *International Journal of Control*, 77(16):1438–1445, Nov 2004.

[351] Alireza Rezaee. Model predictive controller for mobile robot. *Transactions on Environment and Electrical Engineering*, 2(2):18, Jun 2017.

[352] O. Aarna. Dynamic chemical plant model and its application. *IFAC Proceedings Volumes*, 11(1):279 – 286, 1978. 7th Triennial World Congress of the IFAC on A Link Between Science and Applications of Automatic Control, Helsinki, Finland, 12-16 June.

[353] Frederic Gabern, Wang S Koon, Jerrold E Marsden, and Shane D Ross. Theory and computation of non-RRKM lifetime distributions and rates in chemical systems with three or more degrees of freedom. *Physica D: Nonlinear Phenomena*, 211(3-4):391–406, 2005.

[354] Kevin Arulmaran and Jinfeng Liu. Handling model plant mismatch in state estimation using a multiple-model-based approach. *Industrial & Engineering Chemistry Research*, 56(18):5339–5351, Apr 2017.

[355] O. Nelles. *Nonlinear system identification: From classical approaches to neural networks and fuzzy models*. Springer, 2013.

[356] Roger W Brockett. Volterra series and geometric control theory. *Automatica*, 12(2):167–176, 1976.

[357] Bryon R Maner, F. J. Doyle, Babatunde A Ogunnaike, and Ronald K Pearson. A nonlinear model predictive control scheme using second order Volterra models. In *American Control Conference, 1994*, volume 3, pages 3253–3257, 1994.

[358] Richard Lippmann. An introduction to computing with neural nets. *IEEE Assp magazine*, 4(2):4–22, 1987.

[359] Andreas Draeger, Sebastian Engell, and Horst Ranke. Model predictive control using neural networks. *IEEE Control Systems Magazine*, 15(5):61–66, 1995.

[360] Jose C Principe, Ludong Wang, and Mark A Motter. Local dynamic modeling with self-organizing maps and applications to nonlinear system identification and control. *Proceedings of the IEEE*, 86(11):2240–2258, 1998.

[361] Jonathan Ko, Daniel J Klein, Dieter Fox, and Dirk Haehnel. Gaussian processes and reinforcement learning for identification and control of an autonomous blimp. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 742–747. IEEE, 2007.

[362] Jerry A. McMahan and Ralph C. Smith. Parameter estimation for predictive simulation of oscillatory systems with model discrepancy. *IFAC-PapersOnLine*, 49(18):428 – 433, 2016. 10th IFAC Symposium on Nonlinear Control Systems NOLCOS 2016.

[363] You Ling, Joshua Mullins, and Sankaran Mahadevan. Selection of model discrepancy priors in Bayesian calibration. *Journal of Computational Physics*, 276:665 – 680, 2014.

[364] Jenný Brynjarsdóttir and Anthony O'Hagan. Learning about physical parameters: The importance of model discrepancy. *Inverse Problems*, 30(11):114007, Oct 2014.

[365] John A. Burns, Eugene M. Cliff, and Kasie Farlow. Parameter estimation and model discrepancy in control systems with delays. *IFAC Proceedings Volumes*, 47(3):11679 – 11684, 2014. 19th IFAC World Congress.

[366] Haoyang Deng and Toshiyuki Ohtsuka. A highly parallelizable Newton-type method for nonlinear model predictive control. *IFAC-PapersOnLine*, 51(20):349 – 355, 2018. 6th IFAC Conference on Nonlinear Model Predictive Control NMPC 2018.

[367] Moshe Gitterman. *The Chaotic Pendulum*. World Scientific, 2010.

[368] Galileo Galilei. *Dialogue concerning the two chief world systems*. University of California press, 1967.

[369] Galileo Galilei. *Dialogues concerning two new sciences*. Dover, 1914.

[370] Paolo Palmieri. Galileo's experiments with pendulums: Then and now. *PhilSci Archive*, 2007.

[371] Gregory L Baker and James A Blackburn. *The pendulum: A case study in physics*. OUP Oxford, 2008.

[372] Luis T Aguilar, José A Ortega, and Alejandra Ferreira. Self-excited oscillations in an inverted cart–pendulum based on the two-relay approach. *ISA transactions*, 121:306–315, 2022.

[373] Hiroya Oka, Yuji Maruki, Haruo Suemitsu, and Takami Matsuo. Nonlinear control for rotational movement of cart-pendulum system using homoclinic orbit. *International Journal of Control, Automation and Systems*, 14(5):1270–1279, 2016.

[374] Michael R Matthews, Colin F Gauld, and Arthur Stinner. *The pendulum: Scientific, historical, philosophical and educational perspectives*. Springer Science & Business Media, 2005.

[375] Leonid B Freidovich, Anton S Shiriaev, and Ian R Manchester. Experimental implementation of stable oscillations of the Furuta pendulum around the upward equilibrium. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 171–176. IEEE, 2007.

[376] JA Acosta. Furuta's pendulum: A conservative nonlinear model for theory and practise. *Mathematical Problems in Engineering*, 2010, 2010.

[377] Daniel J Block, Karl J Åström, and Mark W Spong. The reaction wheel pendulum. *Synthesis Lectures on Control and mechatronics*, 1(1):1–105, 2007.

[378] Leonid B Freidovich, Pedro La Hera, Uwe Mettin, Anders Robertsson, Anton S Shiriaev, and Rolf Johansson. Shaping stable periodic motions of inertia wheel pendulum: Theory and experiment. *Asian journal of control*, 11(5):548–556, 2009.

[379] Anton S Shiriaev, Leonid B Freidovich, and Sergei V Gusev. Transverse linearization for controlled mechanical systems with several passive degrees of freedom. *IEEE Transactions on Automatic Control*, 55(4):893–906, 2010.

[380] Michael Meindl, Dustin Lehmann, and Thomas Seel. Bridging reinforcement learning and iterative learning control: Autonomous reference tracking for unknown, nonlinear dynamics. *TechRxiv*, 2021.

[381] Daniel Bernoulli. Theoremata de oscillationibus corporum filo flexili connexorum et catenae verticaliter suspensae. *Comm. Acad. Sci. Petrop*, 6(1732-1733):108–122, 1738.

[382] David D. Nolte. The ups and downs of the compound double pendulum, Dec 2020.

[383] L. Euler and C. Truesdell. *The Rational Mechanics of Flexible Or Elastic Bodies 1638 - 1788: Introduction to Vol. X and XI*. Leonhard Euler, Opera Omnia. Springer Basel AG, 1980.

[384] Troy Shinbrot, Celso Grebogi, Jack Wisdom, and James A Yorke. Chaos in a double pendulum. *American Journal of Physics*, 60(6):491–499, 1992.

[385] Joe Chen. Chaos from simplicity: An introduction to the double pendulum, 2008.

[386] MZ Rafat, MS Wheatland, and TR Bedding. Dynamics of a double pendulum with distributed mass. *American Journal of Physics*, 77(3):216–223, 2009.

[387] Tomasz Stachowiak and Toshio Okada. A numerical analysis of chaos in the double pendulum. *Chaos, Solitons & Fractals*, 29(2):417–422, 2006.

[388] Haruo Yoshida. Construction of higher order symplectic integrators. *Physics letters A*, 150(5-7):262–268, 1990.

[389] Jerrold E Marsden and Matthew West. Discrete mechanics and variational integrators. *Acta Numerica*, 10:357–514, 2001.

[390] Julia Timmermann, S Khatab, Sina Ober-Blöbaum, and Ansgar Trächtler. Discrete mechanics and optimal control and its application to a double pendulum on a cart. *IFAC Proceedings Volumes*, 44(1):10199–10206, 2011.

[391] Zeguo Wang, Leonid B Freidovich, and Honghua Zhang. Periodic motion planning and control for double rotary pendulum via virtual holonomic constraints. *IEEE/CAA Journal of Automatica Sinica*, 6(1):291–298, 2017.

[392] Zeguo Wang, Leonid B Freidovich, and Honghua Zhang. Almost periodic motion planning and control for double rotary pendulum with experimental validation. *Asian Journal of Control*, 22(6):2434–2443, 2020.

[393] Karl J Åström, Katsuhisa Furuta, Masafumi Iwashiro, and Tasuku Hoshino. Energy based strategies for swinging up a double pendulum. *IFAC Proceedings Volumes*, 32(2):6450–6455, 1999.

[394] Mark W Spong. The swing up control problem for the Acrobot. *IEEE control systems magazine*, 15(1):49–55, 1995.

[395] Isabelle Fantoni, Rogelio Lozano, and Mark W Spong. Energy based control of the Pendubot. *IEEE Transactions on Automatic Control*, 45(4):725–729, 2000.

[396] Leonid Freidovich, Anders Robertsson, Anton Shiriaev, and Rolf Johansson. Periodic motions of the Pendubot via virtual holonomic constraints: Theory and experiments. *Automatica*, 44(3):785–791, 2008.

[397] Mohd Sazli Saad, Luqman Nul Hakim Mat Deri, Z Shayfull, SM Nasir, and M Fathullah. Parameter estimation of damped compound pendulum using Bat algorithm. In *MATEC Web of Conferences*, volume 78, page 01118. EDP Sciences, 2016.

[398] Marc Peter Deisenroth. *Efficient reinforcement learning using Gaussian processes*, volume 9. KIT Scientific Publishing, 2010.

[399] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[400] Joshua William Burby, Qi Tang, and R Maulik. Fast neural Poincaré maps for toroidal magnetic fields. *Plasma Physics and Controlled Fusion*, 63(2):024001, 2020.

[401] Kyosuke Ono, Koji Yamamoto, and Atushi Imadu. Control of giant swing motion of a two-link horizontal bar gymnastic robot. *Advanced Robotics*, 15(4):449–465, 2001.

[402] Liyana Ramli, Zaharuddin Mohamed, Auwalu M Abdullahi, Hazriq Izzuan Jaafar,

and Izzuddin M Lazim. Control strategies for crane systems: A comprehensive review. *Mechanical Systems and Signal Processing*, 95:1–23, 2017.

[403] Pietro Morasso, Amel Cherif, and Jacopo Zenzeri. Quiet standing: The single inverted pendulum model is not so bad after all. *PloS one*, 14(3):e0213870, 2019.

[404] John EA Bertram and Young-Hui Chang. Mechanical energy oscillations of two brachiation gaits: Measurement and simulation. *American Journal of Physical Anthropology: The Official Publication of the American Association of Physical Anthropologists*, 115(4):319–326, 2001.

[405] Pedro X La Hera, Leonid B Freidovich, Anton S Shiriaev, and Uwe Mettin. New approach for swinging up the Furuta pendulum: Theory and experiments. *Mechatronics*, 19(8):1240–1250, 2009.

[406] Karl Johan Åström and Katsuhisa Furuta. Swinging up a pendulum by energy control. *Automatica*, 36(2):287–295, 2000.

[407] David Kennedy and James Conlon. Inverted pendulum swing up controller, 2011.

[408] Toshiaki Takahashi, Katsuhisa Furuta, Shoshiro Hatakeyama, Satoshi Suzuki, and Akihiko Sugiki. Swing-up control of inverted pendulum by periodic input. *IFAC Proceedings Volumes*, 35(1):283–286, 2002.

[409] K. Furuta, M. Yamakita, and S. Kobayashi. Swing up control of inverted pendulum. In *Proceedings IECON '91: 1991 International Conference on Industrial Electronics, Control and Instrumentation*, pages 2193–2198 vol.3, 1991.

[410] Ioannis Kapnisakis and Catarinacci Francesco. Swing up control of inverted pendulum. Master's thesis, Aalborg University, Oct 2016.

[411] Noriko Matsuda, Masaki Izutsu, Jun Ishikawa, Katsuhisa Furuta, and Karl J Astrom.

Swinging-up and stabilization control based on natural frequency for pendulum systems. In *2009 American Control Conference*, pages 5291–5296. IEEE, 2009.

[412] Adam Mills, Adrian Wills, and Brett Ninness. Nonlinear model predictive control of an inverted pendulum. In *2009 American control conference*, pages 2335–2340. IEEE, 2009.

[413] Tomohide Maeba, Mingcong Deng, Akira Yanou, and Tomohiro Henmi. Swing-up controller design for inverted pendulum by using energy control method based on Lyapunov function. In *Proceedings of the 2010 International Conference on Modelling, Identification and Control*, pages 768–773. IEEE, 2010.

[414] Kazunobu Yoshida. Swing-up control of an inverted pendulum by energy-based methods. In *Proceedings of the 1999 American Control Conference (Cat. No. 99CH36251)*, volume 6, pages 4045–4047. IEEE, 1999.

[415] Knut Graichen and Michael Zeitz. Feedforward control design for finite-time transition problems of nonlinear systems with input and output constraints. *IEEE Transactions on Automatic Control*, 53(5):1273–1278, 2008.

[416] Ernesto Aranda-Escolástico, María Guinaldo, Matilde Santos, and Sebastián Dormido. Control of a chain pendulum: A fuzzy logic approach. *International journal of computational intelligence systems*, 9(2):281–295, 2016.

[417] James Driver and Dylan Thorpe. Design, build and control of a single/double rotational inverted pendulum. *The University of Adelaide, School of Mechanical Engineering, Australia*, 4, 2004.

[418] Yutao Chen, Mattia Bruschetta, Davide Cuccato, and Alessandro Beghi. An adaptive partial sensitivity updating scheme for fast nonlinear model predictive control. *IEEE Transactions on Automatic Control*, 64(7):2712–2726, 2018.

[419] Wei Zhong and Helmut Rock. Energy and passivity based control of the double inverted pendulum on a cart. In *Proceedings of the 2001 IEEE International Conference on Control Applications (CCA'01)(Cat. No. 01CH37204)*, pages 896–901. IEEE, 2001.

[420] Chin Wang Tao, Jinshiuh Taur, JH Chang, and Shun-Feng Su. Adaptive fuzzy switched swing-up and sliding control for the double-pendulum-and-cart system. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 40(1):241–252, 2009.

[421] Tomohiro Henmi, Mingcong Deng, and Akira Inoue. Unified method for swing-up control of double inverted pendulum systems. In *Proceedings of the 2014 International Conference on Advanced Mechatronic Systems*, pages 572–577. IEEE, 2014.

[422] Venkata Dhruva Pamulaparthy. *An Evolutionary Model-Free Controller and its Application to the Swing-Up of a Double Inverted Pendulum*. California State University, Long Beach, 2017.

[423] X Xin. Analysis of the energy-based swing-up control for the double pendulum on a cart. *International journal of robust and nonlinear control*, 21(4):387–403, 2011.

[424] Pathompong Jaiwat and Toshiyuki Ohtsuka. Real-time swing-up of double inverted pendulum by nonlinear model predictive control. In *5th International Symposium on Advanced Control of Industrial Processes*, pages 290–295, 2014.

[425] J Rubı, A Rubio, and A Avello. Swing-up control problem for a self-erecting double inverted pendulum. *IEE Proceedings-Control Theory and Applications*, 149(2):169–175, 2002.

[426] Ke Xu, Julia Timmermann, and Ansgar Trächtler. Swing-up of the moving double pendulum on a cart with simulation based LQR-Trees. *IFAC-PapersOnLine*, 50(1):4094–4100, 2017.

[427] Shinji Nakayama, Mingcong Deng, Shuhui Bi, and Akira Yanou. Genetic algorithm with a robust solution searching scheme based controller parameters selection of a cart-type inverted pendulum. In *Proceedings of the 2010 International Conference on Modelling, Identification and Control*, pages 568–573. IEEE, 2010.

[428] Tobias Glück, Andreas Eder, and Andreas Kugi. Swing-up control of a triple pendulum on a cart with experimental validation. *Automatica*, 49(3):801 – 808, 2013.

[429] Vlastimil Šetka, Roman Čečil, and Miloš Schlegel. Triple inverted pendulum system implementation using a new ARM/FPGA control platform. In *2017 18th International Carpathian Control Conference (ICCC)*, pages 321–326. IEEE, 2017.

[430] Dasheng Liu and Hiroshi Yamaura. Stabilization control for giant swing motions of 3-link horizontal bar gymnastic robot using multiple-prediction delayed feedback control with a periodic gain. *Journal of System Design and Dynamics*, 5(1):42–54, 2011.

[431] Khaled Gamal Eltohamy and Chen-Yuan Kuo. Real time stabilisation of a triple link inverted pendulum using single control input. *IEE Proceedings-Control Theory and Applications*, 144(5):498–504, 1997.

[432] GA Medrano-Cerda. Robust stabilization of a triple inverted pendulum-cart. *International Journal of Control*, 68(4):849–866, 1997.

[433] Vassilios A Tsachouridis. Robust control of a triple inverted pendulum. In *Proceedings of the 1999 IEEE International Conference on Control Applications (Cat. No. 99CH36328)*, volume 2, pages 1235–1240. IEEE, 1999.

[434] Albert CJ Luo and Chuan Guo. Period motions in a periodically forced, damped double pendulum. In *ASME International Mechanical Engineering Congress and Exposition*, volume 52040, page V04BT06A029. American Society of Mechanical Engineers, 2018.

[435] Jaume Llibre, Douglas Duarte Novaes, and Marco Antonio Teixeira. On the periodic solutions of a perturbed double pendulum. *arXiv preprint arXiv:1109.6378*, 2011.

[436] Benjamin Jahn, Lars Watermann, and Johann Reger. On the design of stable periodic orbits of a triple pendulum on a cart with experimental validation. *Automatica*, 125:109403, 2021.

[437] Wang Sang Koon, Martin W Lo, Jerrold E Marsden, and Shane D Ross. Heteroclinic connections between periodic orbits and resonance transitions in celestial mechanics. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 10(2):427–469, 2000.

[438] Gerard Gómez, Wang S Koon, MW Lo, Jerrold E Marsden, Josep Masdemont, and Shane D Ross. Connecting orbits and invariant manifolds in the spatial restricted three-body problem. *Nonlinearity*, 17(5):1571, 2004.

[439] Wang Sang Koon, Martin W Lo, Jerrold E Marsden, and Shane D Ross. Dynamical systems, the three-body problem and space mission design. In *Equadiff 99: (In 2 Volumes)*, pages 1167–1181. World Scientific, 2000.

[440] Swarnendu Mandal, Sudeshna Sinha, and Manish Dev Shrimali. Machine learning potential of a single pendulum. *arXiv preprint arXiv:2201.13390*, 2022.

[441] Allan Sorensen and Anton S Shiriaev. Friction compensation in the Furuta pendulum for stabilizing rotational modes. In *Proceedings of the 40th IEEE Conference on Decision and Control (Cat. No. 01CH37228)*, volume 4, pages 3772–3777. IEEE, 2001.

[442] Anton Shiriaev, Leonid Paramonov, Allan Sørensen, and Anders Robertsson. Stabilization of rotational modes for the Furuta pendulum. *IFAC Proceedings Volumes*, 34(6):813–818, 2001.

[443] Benjamin J Knudson. The double pendulum: Construction and exploration. *Physics Department, California Polytechnic State University*, 2012.

[444] Michael Hesse, Julia Timmermann, Eyke Hüllermeier, and Ansgar Trächtler. A reinforcement learning strategy for the swing-up of the double pendulum on a cart. *Procedia Manufacturing*, 24:15–20, 2018.

[445] David J Christini, James J Collins, and Paul S Linsay. Experimental control of high-dimensional chaos: The driven double pendulum. *Physical Review E*, 54(5):4824, 1996.

[446] Audun D Myers, Joshua R Tempelman, David Petrushenko, and Firas A Khasawneh. Low-cost double pendulum for high-quality data collection with open-source video tracking and analysis. *HardwareX*, 8:e00138, 2020.

[447] Knut Graichen, Michael Treuer, and Michael Zeitz. Fast side-stepping of the triple inverted pendulum via constrained nonlinear feedforward control design. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 1096–1101. IEEE, 2005.

[448] Roman Čečil, Vlastimil Šetka, and Miloš Schlegel. Radio module for fast real-time control of inverse triple pendulum. In *2016 3rd International Symposium on Wireless Systems within the Conferences on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS-SWS)*, pages 80–84. IEEE, 2016.

[449] YV Pavan Kumar and Ravikumar Bhimasingu. Alternative hardware-in-the-loop (HIL) setups for real-time simulation and testing of microgrids. In *2016 IEEE 1st International Conference on Power Electronics, Intelligent Control and Energy Systems (ICPEICES)*, pages 1–6. IEEE, 2016.

[450] HIWIN. *mega-fabs D1 Drive User Guide*. HIWIN MIKROSYSTEM CORP, Taichung 40852, Taiwan, 2011.

[451] Kaori Onozaki, Hiroaki Yoshimura, and Shane D. Ross. Tube dynamics and low

energy Earth-Moon transfers in the 4-body system. *Advances in Space Research*, 60(10):2117–2132, 2017.

[452] M Lo and S Ross. Surfing the solar system: Invariant manifolds and the dynamics of the solar system. *JPL IOM*, 312(97):2–4, 1997.

[453] Jerrold Marsden and Shane Ross. New methods in celestial mechanics and mission design. *Bulletin of the American Mathematical Society*, 43(1):43–73, 2006.

[454] Maxime Murray and JD Mireles James. Homoclinic dynamics in a spatial restricted four-body problem: Blue skies into Smale horseshoes for vertical Lyapunov families. *Celestial Mechanics and Dynamical Astronomy*, 132(6):1–44, 2020.

[455] J-B Caillau, Bilel Daoud, and Joseph Gergaud. Minimum fuel control of the planar circular restricted three-body problem. *Celestial Mechanics and Dynamical Astronomy*, 114(1):137–150, 2012.

[456] Carlos de la Fuente Marcos, Raúl de la Fuente Marcos, and Sverre J Aarseth. Dynamical impact of the Planet Nine scenario: N-body experiments. *Monthly Notices of the Royal Astronomical Society: Letters*, 460(1):L123–L127, 2016.

[457] Sam Hadden, Gongjie Li, Matthew J Payne, and Matthew J Holman. Chaotic dynamics of trans-Neptunian objects perturbed by planet nine. *The Astronomical Journal*, 155(6):249, 2018.

[458] JC Becker, Tali Khain, Stephanie J Hamilton, FC Adams, David W Gerdes, L Zullo, K Franson, S Millholland, GM Bernstein, M Sako, et al. Discovery and dynamical analysis of an extreme trans-Neptunian object with a high orbital inclination. *The Astronomical Journal*, 156(2):81, 2018.

[459] Zdzislaw E Musielak and Billy Quarles. The three-body problem. *Reports on Progress in Physics*, 77(6):065901, 2014.

[460] Jürgen Moser. On the generalization of a theorem of A. Liapounoff. *Communications on Pure and Applied Mathematics*, 11(2):257–271, 1958.

[461] S. Wiggins, L. Wiesenfeld, C. Jaffé, and T. Uzer. Impenetrable barriers in phase-space. *Phys. Rev. Lett.*, 86:5478–5481, Jun 2001.

[462] Marian Gidea and Josep J Masdemont. Geometry of homoclinic connections in a planar circular restricted three-body problem. *International journal of bifurcation and chaos*, 17(04):1151–1169, 2007.

[463] Richard. McGehee. *Some homoclinic orbits for the restricted three-body problem.* The University of Wisconsin-Madison, 1969.

[464] C. C. Conley. Low energy transit orbits in the restricted three-body problems. *SIAM Journal on Applied Mathematics*, 16(4):732–746, July 1968.

[465] Josep J Masdemont. A review of invariant manifold dynamics of the CRTBP and some applications. *Nonlinear science and complexity*, pages 139–146, 2011.

[466] Shane D Ross. The interplanetary transport network: Some mathematical sophistication allows spacecraft to be maneuvered over large distances using little or no fuel. *American Scientist*, 94(3):230–237, 2006.

[467] G. Gómez, Fundació Bosch i Gimpera. Barcelona, and European Space Agency. *Study Refinement of Semi-analytical Halo Orbit Theory: Executive Summary ; ESOC Contract No.: 8625/89/D/MD (SC)*. Fundació Bosch i Gimpera, 1991.

[468] Jaume Llibre, Regina Martínez, and Carles Simó. Tranversality of the invariant manifolds associated to the Lyapunov family of periodic orbits near L2 in the restricted three-body problem. *Journal of Differential Equations*, 58(1):104–156, 1985.

[469] Hai-Jun Liao, Jin-Guo Liu, Lei Wang, and Tao Xiang. Differentiable programming tensor networks. *Physical Review X*, 9(3):031041, 2019.

[470] Stephen Smale. Differentiable dynamical systems. *Bulletin of the American mathematical Society*, 73(6):747–817, 1967.

[471] HIWIN. *Linear Motor System Technical Information*. HIWIN MIKROSYSTEM CORP, Taichung 40852, Taiwan, 2017.

[472] Vladimir Kraz. Mitigating EMI issues in servo motors and variable frequency drives. *2016 EMC Test and Design Guide*, pages 9–16, 2016.

Appendix A

APPENDIX FOR CHAPTER 3

A.1 Noise Sensitivity of SINDy-PI and implicit-SINDy

A.1.1 Performance Evaluation Criteria

To compare the performance of SINDy-PI and implicit-SINDy for noisy data, we must define an evaluation criteria. We compare the performance of the best model generated by each method that has the lowest prediction error on the test data, selected according to Eq. (3.7). To compare the models generated by the two methods with the ground truth model, we use the concept of model discrepancy [178, 347, 348] and set prediction accuracy, structural accuracy, and parameter accuracy as our performance criteria. A good prediction error does not guarantee the model has good structural accuracy and parameter accuracy, and vice versa, motivating multiple performance criteria.

A.1.2 Numerical Experiments

We use the Michaelis–Menten kinetics, given by Eq. (3.12), to compare the performance of SINDy-PI and implicit-SINDy. We performed our numerical experiments as follows:

Step 1: Randomly generate 2400 different initial conditions of different magnitudes ranging from 0 to 12.5. Simulate those initial conditions using a fourth-order Runge-Kutta method with time step $dt = 0.1$ and time horizon $T = 5$. The testing data is generated using 600 random initial conditions using the same method as the training data.

Step 2: Add Gaussian noise to the training and testing data. 23 different Gaussian noise

levels with magnitudes ranging from 10^{-7} to 5×10^{-1} are used. For each noise level, 30 different random noise realizations are generated, resulting in 30 different noisy data sets for each noise level.

Step 3: Compute the derivative of the noisy data. We investigate several approaches, including finite-difference and total-variation regularized difference (TVRegDiff) [310] derivatives. In all cases, SINDy-PI is several orders of magnitude more robust to noise than implicit-SINDy, and only the result of using TVRegDiff is shown in this paper. TVRegDiff generates more accurate derivatives, but also requires hyperparameter tuning and causes aliasing, so we trim the ends of the time series generated by each initial condition (first and last 30%). It is possible to add Gaussian noise to the clean derivative data to investigate robustness, although this is less relevant for real-world scenarios, where only noisy state data is available.

Step 4: Train SINDy-PI and implicit-SINDy models on noisy training data. For each noise level, we sweep through 68 different sparsity parameters λ for SINDy-PI, from 0.01 to 5. The λ is varied by a factor of 2 [182] to calculate the null space in the implicit-SINDy method. The library for implicit-SINDy and SINDy-PI is

$$\boldsymbol{\Theta}(\mathbf{X}, \dot{\mathbf{X}}) = [\mathbf{1}\ \mathbf{X}\ \mathbf{X}^2\ \mathbf{X}^3\ \mathbf{X}^4\ \dot{\mathbf{X}}\ \dot{\mathbf{X}}\mathbf{X}\ \dot{\mathbf{X}}\mathbf{X}^2\ \dot{\mathbf{X}}\mathbf{X}^3\ \dot{\mathbf{X}}\mathbf{X}^4]. \tag{A.1}$$

Step 5: Due to the various parameter values, we use model selection to choose a model. We use the test data with the same noise magnitude to perform the model selection process. The ratio of training data and testing data is $8:2$.

Step 6: The best model generated by the two methods are compared. We use the prediction error, error in the model structure (i.e., the number of terms that are incorrectly present or missing from the model), and parameter error as our model performance

evaluation criteria. We average the performance over 30 different noise realizations for each noise level. We then plot the distribution of structure error in Fig. 3.3.

Many parameters affect the performance of these methods: the length of training data, prediction steps to calculate prediction error, the initial conditions for training data, choice of the library, and the derivative computation. We have attempted to carefully optimize each method, although an exhaustive parameter sweep is beyond the scope of the present work. However, in all cases SINDy-PI outperforms implicit-SINDy.

A.2 Data Usage of SINDy-PI and implicit-SINDy

Sec. 3.2.4 investigates the data usage of SINDy-PI and implicit-SINDy on the yeast glycolysis model in Eq. (3.13). The parameters of this problems are given in Table. A.1. The data usage comparison is performed by the following steps:

Step 1: Generate training data by simulating Eq. (3.13) with parameters in Table. A.1 and a time step of $dt = 0.1$, with time horizon $T = 5$. We simulate the system using 900 random initial conditions with magnitude ranging from 0 to 3.

Step 2: Shuffle the training data and select j percent of the entire training data set at random to train the SINDy-PI and implicit-SINDy models. These training data are sampled from all trajectories, and they are not necessarily consecutive in time. No noise is added since we only care about the effect of the data length in this case. The sparsity parameter λ is fixed for both algorithms (different values); this value is selected for a single percentage j where both methods fail to identify the correct model, and we sweep through λ.

Step 3: Run the numerical experiment for 20 times for each data length and calculate the percentage of times the two algorithms yield the correct structure of the Eq. (3.13f).

The final comparison is shown in Fig. 3.4. Data usage requirements for other state equations are given in Table A.2; Fig. 3.4 shows results for the hardest equation to identify.

Table A.1: The parameter used for simulating the Eq. (3.13).

Parameter	c_1	c_2	c_3	d_1	d_2	d_3	d_4	e_1	e_2	e_3	e_4	f_1	f_2
Value	2.5	-100	13.6769	200	13.6769	-6	-6	6	-64	6	16	64	-13
Parameter	f_3	f_4	f_5	g_1	g_2	h_1	h_2	h_3	h_4	h_5	j_1	j_2	j_3
Value	13	-16	-100	1.3	-3.1	-200	13.6769	128	-1.28	-32	6	-18	-100

Table A.2: Comparison of data usage of SINDy-PI and implicit-SINDy on other states.

	Equation	Eq. (3.13a)		Eq. (3.13b)		Eq. (3.13c)	Eq. (3.13d)	Eq. (3.13e)	Eq. (3.13f)		Eq. (3.13g)
	Library Order	6		6		3	3	3	6		3
SINDy-PI un-normalized	Left-Hand Side	$\dot{x}_1$	$\dot{x}_1x_6^4$	$\dot{x}_2$	$\dot{x}_2x_6^4$	$\dot{x}_3$	$\dot{x}_4$	$\dot{x}_5$	$\dot{x}_6x_4^6$		$\dot{x}_7$
	Threshold	0.5	0.05	0.5	0.05	0.2	0.5	0.3	0.01		0.5
	Data Usage	50%	7.5%	55%	8.5%	0.5%	0.5%	0.3%	40%		0.5%
SINDy-PI normalized	Left-Hand Side	$\dot{x}_1$	$\dot{x}_1x_6^4$	$\dot{x}_2$	$\dot{x}_2x_6^4$	$\dot{x}_3$	$\dot{x}_4$	$\dot{x}_5$	$\dot{x}_6$	$\dot{x}_6x_4^6$	$\dot{x}_7$
	Threshold	0.5	0.5	0.5	0.5	0.6	0.8	0.4	0.1	0.1	0.2
	Data Usage	18%	3%	10%	3%	0.45%	0.45%	0.275%	35%	8%	0.4%
implicit-SINDy normalized	Threshold	5×10^{-3}		2×10^{-3}		8×10^{-3}	8×10^{-3}	8×10^{-3}	3×10^{-3}		8×10^{-3}
	Data Usage	10%		10%		0.5%	0.6%	0.3%	100%		0.5%

The other equations require less data. Normalizing the SINDy-PI library improves data learning rates as well.

A.3 SINDy-PI and PDE-FIND on Rational PDE Problem

In Sec. 3.2.5, we compared the performance of SINDy-PI and PDE-FIND on a modified KdV equation. The simulation data is obtained using a spectral method [317] with a time step of $dt = 0.01$ and time horizon $T = 20$, spatial domain $L = -25$ to 25, and $n = 128$ spatial discretization points. The library of PDE-FIND is chosen to be

$$\begin{aligned}\boldsymbol{\Theta}(\mathbf{U}, \mathbf{U}_x, \mathbf{U}_{xx}, \mathbf{U}_{xxx}) =&[\mathbf{1}\ \mathbf{U}\ \mathbf{U}_x\ \mathbf{U}_{xx}\ \mathbf{U}_{xxx}\ \mathbf{U}_x^2\ \mathbf{U}_{xx}^2\ \mathbf{U}_{xxx}^2\ \mathbf{U}\mathbf{U}_x \\ &\mathbf{U}\mathbf{U}_{xx}\ \mathbf{U}\mathbf{U}_{xxx}\ \mathbf{U}^2\mathbf{U}_x^2\ \mathbf{U}^2\mathbf{U}_{xx}^2\ \mathbf{U}^2\mathbf{U}_{xxx}^2]\end{aligned} \tag{A.2}$$

and the right-hand side library for the SINDy-PI is chosen to be

$$\begin{aligned}\boldsymbol{\Theta}(\mathbf{U}, \mathbf{U}_x, \mathbf{U}_{xx}, \mathbf{U}_{xxx}) =&[\mathbf{1}\ \mathbf{U}\ \mathbf{U}_t\ \mathbf{U}_x\ \mathbf{U}_{xx}\ \mathbf{U}_{xxx}\ \mathbf{U}^2\ \mathbf{U}_t^2\ \mathbf{U}_x^2\ \mathbf{U}_{xx}^2\ \mathbf{U}_{xxx}^2 \\ &\mathbf{U}\mathbf{U}_t\ \mathbf{U}\mathbf{U}_x\ \mathbf{U}\mathbf{U}_{xx}\ \mathbf{U}\mathbf{U}_{xxx}\ \mathbf{U}^2\mathbf{U}_x^2\ \mathbf{U}^2\mathbf{U}_{xx}^2\ \mathbf{U}^2\mathbf{U}_{xxx}^2],\end{aligned} \tag{A.3}$$

Table A.3: Parameters used to simulate the double pendulum.

Parameter	m_1	m_2	L_1	L_2	a_1	a_2	I_1	I_2	g
Value	0.2704	0.2056	0.2667	0.2667	0.191	0.1621	0.003	0.0011	9.81

Table A.4: Parameters used to simulate the single pendulum on a cart.

Parameter	m	L	M	g
Value	1	1	1	9.81

Table A.5: Parameters Used in Eq. (3.16) for Simulating the Belousov-Zhabotinsky Reaction Model.

Parameter	q	f	ε	α	β	γ	ε_2	ε_3	χ	D_x	D_z	D_s	D_u
Value	1	1.5	0.3	0.3	0.26	0.4	0.15	0.03	0	0.01	0.01	1	1

while the left-hand side library is chosen to be

$$C(\mathbf{U}, \mathbf{U}_t, \mathbf{U}_x, \mathbf{U}_{xx}) = [\mathbf{U}_t \ \mathbf{U}\mathbf{U}_t \ \mathbf{U}\mathbf{U}_x \ \mathbf{U}\mathbf{U}_{xx}]. \tag{A.4}$$

For both SINDy-PI and PDE-FIND, we used 100 different values for the sparsity parameter λ ranging from 0.1 to 10 with step size 0.1. We use 80% of the simulation data for training and 20% for testing and model selection. We calculate the normalized prediction error for all models on state u_t and the model with minimum prediction error is selected as the final model.

A.4 Parameter Values for Simulations

The parameters for the double pendulum simulation in Sec. 3.3.1 are given in Table. A.3. The parameters used to simulate the simplified model of the Belousov-Zhabotinsky reaction in Eq. (3.16) are given in Table. A.5.

A.5 SINDy-PI Models for the Single Pendulum on a Cart

The Lagrangian for the single pendulum on a cart with an input force on the cart is:

$$\mathcal{L} = T - V = \frac{1}{2}(m+M)\dot{s}^2 + \frac{1}{2}L^2 m\dot{\phi}^2 - Lgm\cos(\phi) + Lm\cos(\phi)\dot{\phi}\dot{s}, \tag{A.5}$$

where m is the mass at the end of the pendulum arm, M is the mass of the cart, L is the length of the pendulum arm, s is the position of the cart, and ϕ is the pendulum angle. We do not consider damping in this case. Using the numeric values $m = M = L = 1$ and $g = -9.81$ this simplifies to

$$\mathcal{L} = T - V = \dot{s}^2 + \frac{1}{2}\dot{\phi}^2 - 9.81\cos(\phi) + \cos(\phi)\dot{\phi}\dot{s}, \tag{A.6}$$

The Euler-Lagrange equation of the system are

$$\begin{aligned} \frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \dot{\phi}} - \frac{\partial \mathcal{L}}{\partial \phi} &= 0, \\ \frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \dot{s}} - \frac{\partial \mathcal{L}}{\partial s} &= F, \end{aligned} \quad \Longrightarrow \quad \begin{aligned} mL^2\ddot{\phi} + mL\ddot{s}\cos(\phi) - Lgm\sin(\phi) &= 0 \\ (M+m)\ddot{s} - F - mL\sin(\phi)\dot{\phi}^2 + mL\ddot{\phi}\cos(\phi) &= 0 \end{aligned} \tag{A.7a}$$

where F is the force applied to the pendulum cart. It is possible to isolate $\ddot{\phi}$ and $\ddot{s}$:

$$\ddot{\phi} = \frac{-(F\cos(\phi) - Mg\sin(\phi) - mg\sin(\phi) + Lm\cos(\phi)\sin(\phi)\dot{\phi}^2)}{L(M + m\sin(\phi)^2)}, \tag{A.8a}$$

$$\ddot{s} = \frac{F + Lm\sin(\phi)\dot{\phi}^2 - mg\cos(\phi)\sin(\phi)}{M + m\sin(\phi)^2}. \tag{A.8b}$$

It is possible to write this as a system of four coupled first-order equations

$$\frac{d}{dt}\phi = \dot{\phi}, \tag{A.9a}$$
$$\frac{d}{dt}s = \dot{s}, \tag{A.9b}$$
$$\frac{d}{dt}\dot{\phi} = \frac{-(F\cos(\phi) - Mg\sin(\phi) - mg\sin(\phi) + Lm\cos(\phi)\sin(\phi)\dot{\phi}^2)}{L(M + m\sin(\phi)^2)}, \tag{A.9c}$$
$$\frac{d}{dt}\dot{s} = \frac{F + Lm\sin(\phi)\dot{\phi}^2 - mg\cos(\phi)\sin(\phi)}{M + m\sin(\phi)^2}. \tag{A.9d}$$

With the numerical values shown in Table. A.4, this becomes

$$\frac{d}{dt}\phi = \dot{\phi}, \tag{A.10a}$$
$$\frac{d}{dt}s = \dot{s}, \tag{A.10b}$$
$$\frac{d}{dt}\dot{\phi} = \frac{19.62\sin(\phi) - F\cos(\phi) - \sin(\phi)\cos(\phi)\dot{\phi}^2}{2 - \cos(\phi)^2}, \tag{A.10c}$$
$$\frac{d}{dt}\dot{s} = \frac{2F - 9.81\sin(2\phi) + 2\sin(\phi)\dot{\phi}^2}{2 + 2\sin(\phi)^2}. \tag{A.10d}$$

Parameters identified by SINDy-PI under different noise magnitudes are presented in Tables A.6 and A.7.

Table A.6: Parameters identified by SINDy-PI for Eq. (A.10c) under different noise magnitudes.

Value \ Basis / Noise Magnitude	Numerator			Denominator	
	$\sin(\phi)$	$F\cos(\phi)$	$\sin(\phi)\cos(\phi)\dot{\phi}^2$	Constant	$\cos(\phi)^2$
0	19.62	-1	-1	2	-1
0.001	19.618	-1.0005	-0.9999	2	-1
0.005	19.6135	-1.171	-0.9996	2	-0.9997
0.02	19.5881	Not Identified	-0.4912	2	-1.0122

Table A.7: Parameters identified by SINDy-PI for Eq. (A.10d) under different noise magnitudes.

Value \ Basis / Noise Magnitude	Numerator			Denominator	
	F	$\sin(2\phi)$	$\sin(\phi)\dot{\phi}^2$	Constant	$\sin(\phi)^2$
0	2	-9.81	2	2	2
0.001	1.9992	-9.816	1.9992	2	1.9992
0.005	1.9982	-9.8015	1.9986	2	1.9986
0.02	2.0705	-9.8234	2.0041	2	2.0041

A.6 SINDy-PI Models for the Mounted Double Pendulum

For a mounted double pendulum system shown in Fig. 3.6 we could have following parameters: the parameters of the pendulum are center of mass m_1 and m_2, center of mass position a_1 and a_2, arm length L_1 and L_2, arm inertia I_1 and I_2, arm rotational angle ϕ_1 and ϕ_2, gravity acceleration g. Those values could be seen from Table. A.3. If we consider friction between the pendulum joint, we could define $k_1 = 7.2485 \times 10^{-4}$ and $k_2 = 1.6522 \times 10^{-4}$ as our damping coefficient.It is easy to derive the Lagrangian of the mounted double pendulum which is given by

$$\begin{aligned}\mathcal{L} = T - V =&(m_2((L_1\cos(\phi_1)\dot{\phi}_1 + a_2\cos(\phi_2)\dot{\phi}_2)^2 + (L_1\sin(\phi_1)\dot{\phi}_1 + a_2\sin(\phi_2)\dot{\phi}_2)^2))/2 + \\ &(m_1(a_1^2\cos(\phi_1)^2\dot{\phi}_1^2 + a_1^2\sin(\phi_1)^2\dot{\phi}_1^2))/2 + (I_1\dot{\phi}_1^2)/2 + (I_2\dot{\phi}_2^2)/2 - gm_2(a_2\cos(\phi_2) \\ &+ L_1\cos(\phi_1)) - a_1gm_1\cos(\phi_1)\end{aligned} \tag{A.11}$$

The damping term caused by friction with friction coefficients k_1 and k_2 is

$$R_a = \frac{1}{2}k_1\dot{\phi}_1 + \frac{1}{2}k_2(\dot{\phi}_1 - \dot{\phi}_2)^2 \tag{A.12}$$

The Euler-Lagrange equations with a Rayleigh dissipation term are then:

$$\frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \dot{\phi}_1} - \frac{\partial \mathcal{L}}{\partial \phi_1} + \frac{\partial R_a}{\partial \dot{\phi}_1} = 0, \tag{A.13a}$$

$$\frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \dot{\phi}_2} - \frac{\partial \mathcal{L}}{\partial \phi_2} + \frac{\partial R_a}{\partial \dot{\phi}_2} = 0. \tag{A.13b}$$

The symbolic form of the Eq. (A.13a) is

$$\begin{aligned} &I_1\ddot{\phi}_1 + k_1\dot{\phi}_1 + k_2\dot{\phi}_1 + L_1^2\ddot{\phi}_1 m_2 + a_1^2\ddot{\phi}_1 m_1 + L_1 a_2 m_2 \sin(\phi_1 - \phi_2)\dot{\phi}_2^2 \\ &+ L_1 a_2 \ddot{\phi}_2 m_2 \cos(\phi_1 - \phi_2) - k_2\dot{\phi}_2 - L_1 g m_2 \sin(\phi_1) - a_1 g m_1 \sin(\phi_1) = 0, \end{aligned} \tag{A.14}$$

and the symbolic form of Eq. (A.13b) is

$$\begin{aligned} &I_2\ddot{\phi}_2 + k_2\dot{\phi}_2 + a_2^2\ddot{\phi}_2 m_2 + L_1 a_2 \ddot{\phi}_1 m_2 \cos(\phi_1 - \phi_2) - k_2\dot{\phi}_1 \\ &- a_2 g m_2 \sin(\phi_2) - L_1 a_2 m_2 \sin(\phi_1 - \phi_2)\dot{\phi}_1^2 = 0. \end{aligned} \tag{A.15}$$

Using the numerical parameter values in these equations gives

$$\ddot{\phi}_1 + 0.03235\dot{\phi}_1 + 0.323\ddot{\phi}_2 \cos(\phi_1 - \phi_2) + 0.323\dot{\phi}_2^2 \sin(\phi_1 - \phi_2) - 0.006006\dot{\phi}_2 - 37.97 \sin(\phi_1) = 0.$$

$$\ddot{\phi}_2 + 0.02525\dot{\phi}_2 + 1.358\ddot{\phi}_1 \cos(\phi_1 - \phi_2) - 0.02525\dot{\phi}_1 - 49.94 \sin(\phi_2) - 1.358\dot{\phi}_1^2 \sin(\phi_1 - \phi_2) = 0.$$

If we set $k_1 = k_2 = 0$ and combine the equations, it is possible to solve for $\ddot{\phi}_1$ and $\ddot{\phi}_2$

$$\begin{aligned} \ddot{\phi}_1 = &(L_1 a_2^2 g m_2^2 \sin(\phi_1) - 2L_1 a_2^3 \dot{\phi}_2^2 m_2^2 \sin(\phi_1 - \phi_2) + 2I_2 L_1 g m_2 \sin(\phi_1) \\ &+ L_1 a_2^2 g m_2^2 \sin(\phi_1 - 2\phi_2) + 2I_2 a_1 g m_1 \sin(\phi_1) - L_1^2 a_2^2 \dot{\phi}_1^2 m_2^2 \sin(2\phi_1 - 2\phi_2) \\ &- 2I_2 L_1 a_2 \dot{\phi}_2^2 m_2 \sin(\phi_1 - \phi_2) + 2a_1 a_2^2 g m_1 m_2 \sin(\phi_1))/(2I_1 I_2 + L_1^2 a_2^2 m_2^2 \\ &+ 2I_2 L_1^2 m_2 + 2I_2 a_1^2 m_1 + 2I_1 a_2^2 m_2 - L_1^2 a_2^2 m_2^2 \cos(2\phi_1 - 2\phi_2) + 2a_1^2 a_2^2 m_1 m_2) \end{aligned}$$

and

$$\begin{aligned}\ddot{\phi}_2 = (&a_2 m_2(2I_1 g\sin(\phi_2) + 2L_1^3\dot{\phi}_1^2 m_2\sin(\phi_1-\phi_2) + 2L_1^2 g m_2\sin(\phi_2) + 2I_1 L_1\dot{\phi}_1^2\sin(\phi_1-\phi_2)\\ &+ 2a_1^2 g m_1\sin(\phi_2) + L_1^2 a_2\dot{\phi}_2^2 m_2\sin(2\phi_1-2\phi_2) + 2L_1 a_1^2\dot{\phi}_1^2 m_1\sin(\phi_1-\phi_2)\\ &- 2L_1^2 g m_2\cos(\phi_1-\phi_2)\sin(\phi_1) - 2L_1 a_1 g m_1\cos(\phi_1-\phi_2)\sin(\phi_1)))\\ &/(2(I_1I_2 + L_1^2a_2^2m_2^2 + I_2L_1^2m_2 + I_2a_1^2m_1 + I_1a_2^2m_2 - L_1^2a_2^2m_2^2\cos(\phi_1-\phi_2)^2 + a_1^2a_2^2m_1m_2)).\end{aligned}$$

If we use the values in Table. A.3 we have

$$\begin{aligned}\ddot{\phi}_1 = (&-0.2808\sin(2\phi_1-2\phi_2)\dot{\phi}_1^2 - 0.4136\sin(\phi_1-\phi_2)\dot{\phi}_2^2\\ &+ 10.3278\sin(\phi_1-2\phi_2) + 38.2984\sin(\phi_1))/(1-0.2808\cos(2\phi_1-2\phi_2)),\end{aligned}$$

and

$$\begin{aligned}\ddot{\phi}_2 = (&1.7390\sin(\phi_1-\phi_2)\dot{\phi}_1^2 + 0.2808\sin(2\phi_1-2\phi_2)\dot{\phi}_2^2\\ &- 33.02\sin(2\phi_1-\phi_2) + 30.9472\sin(\phi_2))/(1-0.2808\cos(2\phi_1-2\phi_2)).\end{aligned}$$

With no noise, SINDy-PI discovers the correct equations. When we add random noise with magnitude of 0.005, SINDy-PI discovers the following

$$\begin{aligned}\ddot{\phi}_1 = (&-0.2799\sin(2\phi_1-2\phi_2)\dot{\phi}_1^2 - 0.4137\sin(\phi_1-\phi_2)\dot{\phi}_2^2 +\\ &+ 10.3429\sin(\phi_1-2\phi_2) + 38.3117\sin(\phi_1))/(1-0.2815\cos(2\phi_1-2\phi_2)),\end{aligned}$$

and

$$\begin{aligned}\ddot{\phi}_2 = (&1.7392\sin(\phi_1-\phi_2)\dot{\phi}_1^2 + 0.2805\sin(2\phi_1-2\phi_2)\dot{\phi}_2^2\\ &- 33.0035\sin(2\phi_1-\phi_2) + 30.9418\sin(\phi_2))/(1-0.2813\cos(2\phi_1-2\phi_2)).\end{aligned}$$

If we increase the noise magnitude to 0.01 then the SINDy-PI discovered equation

becomes

$$\ddot{\phi}_1 = (-0.2768\dot{\phi}_1^2 \sin(2\phi_1 - 2\phi_2) - 0.4138 \sin(\phi_1 - \phi_2)\dot{\phi}_2^2 + 10.3676 \sin(\phi_1 - 2\phi_2) + 38.3225 \sin(\phi_1))/(1 - 0.2818 \cos(2\phi_1 - 2\phi_2)),$$

and

$$\ddot{\phi}_2 = (1.7355 \sin(\phi_1 - \phi_2)\dot{\phi}_1^2 + 0.2794 \sin(2\phi_1 - 2\phi_2)\dot{\phi}_2^2 + 0.1675 \sin(2\phi_1 - 2\phi_2) - 33.0445 \sin(2\phi_1 - \phi_2) + 31.0065 \sin(\phi_2))/(1 - 0.2819 \cos(2\phi_1 - 2\phi_2)).$$

If we continue to increase the noise magnitude to 0.05 then SINDy-PI incorrectly identifies

$$\begin{aligned}\ddot{\phi}_1 = (&15.5413 \sin(\phi_1 - 2\phi_2) - 2.6396 \sin(\phi_1 - \phi_2) - 0.9538 \cos(\phi_1 - \phi_2) \\ &+ 35.9971 \cos(\phi_1 - 61/40) - 2.4160 \cos(\phi_2 - 1149/1000) - 0.0733 \cos(2\phi_1 - 2\phi_2) \\ &+ 0.0269 \cos(4\phi_1 - 2\phi_2) + 2.3419 \sin(2\phi_1 - \phi_2) + 0.5142\dot{\phi}_1 \sin(2\phi_1 - 2\phi_2) \\ &- 0.4584\dot{\phi}_2 \sin(2\phi_1 - \phi_2) - 0.3807\dot{\phi}_2^2 \sin(\phi_1 - \phi_2) + 0.8810\dot{\phi}_1 \sin(\phi_1 - \phi_2) \\ &+ 0.9411\dot{\phi}_1 \sin(\phi_1 - 2\phi_2) - 0.7664\dot{\phi}_2 \sin(\phi_1 - \phi_2) + 1.2026)/(1 - 0.4245 \cos(2\phi_1 - 2\phi_2)),\end{aligned}$$

and

$$\ddot{\phi}_2 = 70.9 \sin(\phi_1 - \phi_2).$$

A.7 SINDy-PI Model for the Belousov-Zhabotinsky Reaction

The SINDy-PI discovered PDE for the simplified BZ reaction is

$$x_\tau = \Delta x + \frac{0.24667x + 0.33333s + 0.5z + 3.3333xs - 5.0xz + 2.1333x^2 - 3.3333x^3}{x + 0.1},$$
$$z_\tau = 0.01\Delta z + x + 0.4u - 1.3z,$$
$$s_\tau = \Delta s + 0.17333r - 0.66667s,$$
$$u_\tau = \Delta u - 133.33u + 100z.$$

A.8 Inability to Identify Rational Dynamics with SINDy

In this section, we demonstrate that it is not possible to identify rational dynamics with the original SINDy algorithm, testing it on the Michaelis-Menten dynamics in Eq. (3.12). We use the same parameters in Sec. 3.2.3. The Taylor expansion of Eq. (3.12) at $x = 0$ is

$$\dot{x} \approx 0.6 - 5x + \frac{50}{3}x^2 - \frac{500}{9}x^3 + \frac{5000}{27}x^4 - \frac{50000}{81}x^5. \tag{A.16}$$

Thus, when the trajectory provided for training is close enough to $x = 0$, SINDy should identify Eq. (A.16). To verify this, data with $x_0 = 0.2409$ is simulated for 22 time steps with $dt = 0.01$. Both the Implicit-SINDy and SINDy-PI algorithms identify the correct model in Eq. (3.12) with highly accurate parameters. The model identified by SINDy is

$$\dot{x} = 0.5914 - 4.7387x + 13.1389x^2 - 27.6470x^3 + 36.9846x^4 - 22.8388x^5. \tag{A.17}$$

Thus, SINDy correctly identifies the first three terms of the Taylor expansion, although the higher order terms have large parameter errors. This model is compared with the SINDy-PI and Implicit-SINDy models in Fig. A.1 for a test trajectory initialized with $x_0 = 0.6$. From Fig. A.1 (a), it can be seen that both SINDy-PI and Implicit-SINDy match the true solution. However, the SINDy model only agrees for x near the origin. When Gaussian noise of

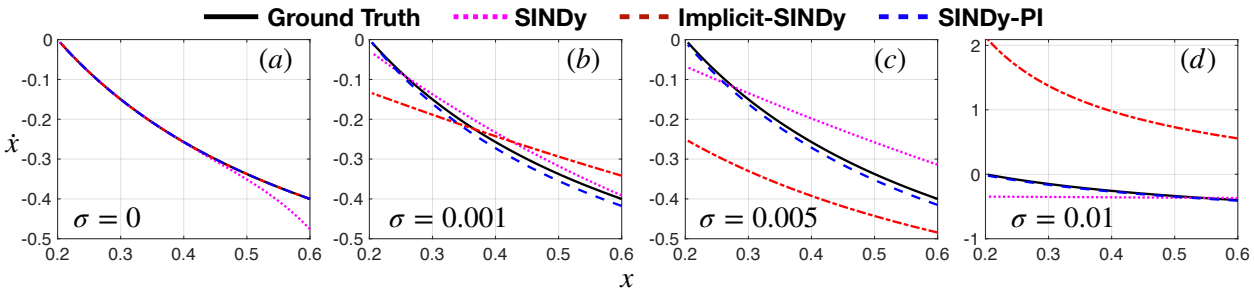


Figure A.1: Comparison of the model identified by SINDy, Implicit-SINDy, and SINDy-PI on Mechaelis-Menten dynamics.

magnitude σ is added, the SINDy model degrades further. Moreover, the amount of data used needs to be increased to identify the correct model. In Fig. A.1 (b) to (d), 330, 2200 and 4400 data points ranging from 0 to 12 are used for training. The same amount of data is used for model selection. From Fig. A.1 (b) to (d), it can be seen that the SINDy model does not work well away from $x = 0$.

Table A.8: The effect of K_m on the noise robustness of SINDy-PI.

K_m	Max Magnitude of Noise	True Model	Identified Model
0.01	0.001	$\dot{x} = -\frac{(0.9x-0.006)}{x+0.01}$	$\dot{x} = -\frac{(x-7.6144)(x-5.6092)(x-2.5078)(x-0.0072)}{(x-7.6146)(x-5.6125)(x-2.5092)(x+0.0092)}$
0.1	0.01	$\dot{x} = -\frac{(0.9x-0.06)}{x+0.1}$	$\dot{x} = -\frac{(x-5.4132)(x-0.0681)}{(x-5.4144)(x+0.099)}$
1	0.05	$\dot{x} = -\frac{(0.9x-0.6)}{x+1}$	$\dot{x} = -\frac{(x-4.7143)(x-0.6726)}{(x-4.7159)(x+0.9698)}$
10	0.1	$\dot{x} = -\frac{(0.9x-6)}{x+10}$	$\dot{x} = -\frac{0.99x-6.583}{x+11.07}$

A.9 Robustness of SINDy and SINDy-PI

The robustness of SINDy-PI to noise depends on a number of factors, including the the length of the data, which initial conditions are chosen to generate the data, the model parameters, the order of the polynomial terms in the model, etc. The relative impact of all of these factors varies with the system we are studying.

In general, training data that explores more of the phase space will result in a more robust model discovery process. Generally speaking, data that results in a better-conditioned $\boldsymbol{\Theta}$ matrix will provide more robust results. Several studies have explored strategies to improve the robustness, with Wu and Xiu suggesting the use of a large ensemble of initial conditions [305]. Exciting transients is also important [180].

Another key factor that affects the condition number of $\boldsymbol{\Theta}$ is the number of library elements, which is determined by the order for a polynomial library. Including higher-order terms increases the condition number, making it more difficult to accurately disambiguate which nonlinear term is responsible for the observed behavior. The library size scales exponentially with the maximum polynomial order.

Noise robustness is also affected by the model parameters. Smaller parameters in $\boldsymbol{\Xi}$ are more likely to be removed during thresholding, making the procedure less robust to noise. To illustrate this, we change the parameter K_m in Eq. (3.12) and test the maximum noise SINDy-PI can handle. We set K_m equal to 0.01, 0.1, 1 and 10 separately. We generate the training data with 120 random initial conditions ranging from 0 to 10. Each initial condition is simulated until $T = 5$ with $dt = 0.01$, and the same magnitude of Gaussian noise is added. TVRegDiff is used to calculate the derivative, and the first and last 30% percent of data is discarded due to aliasing effect. The testing data is generated using the same process and the ratio of training and testing data is $1 : 1$. The model and maximum magnitude of noise allowed for each values of K_m is summarized in Table A.8. These results suggest that as K_m increases, the maximum noise SINDy-PI can handle also increases.

Appendix B

APPENDIX FOR CHAPTER 4

B.1 Algorithm for Simultaneously Denoising and Learning System Model

Algorithm 1: Modified SINDy

Input: $\mathbf{Y}$, $\mathbf{\Theta}(*)$, dt, λ, N_{loop}, ω

Output: $\mathbf{\Xi}$, $\hat{\mathbf{N}}$

/* Initialize the value of $\hat{\mathbf{N}}$ */

if SoftStart **then**

$\hat{\mathbf{N}} = \mathbf{Y} - \text{smoothSignal}(\mathbf{Y})$ // If the soft start is true, the estimated value of noise is obtained by pre-smoothing the noisy signal.

else

$\hat{\mathbf{N}} = \text{zeros}(\text{size}(\mathbf{Y}))$ // Else, the estimated value of noise is initialized using zero matrix.

/* Initialize the value of $\mathbf{\Xi}$ */

$\hat{\mathbf{X}} = \mathbf{Y} - \hat{\mathbf{N}}$.

Calculate $\dot{\hat{\mathbf{X}}}$ using $\hat{\mathbf{X}}$.

$\mathbf{\Xi} = \text{SINDy}(\dot{\hat{\mathbf{X}}}, \mathbf{\Theta}(\hat{\mathbf{X}}), \lambda)$.

/* Simultaneously denoising and learning system model */

while $k < N_{loop}$ **do**

Optimize $\mathcal{L}(\mathbf{\Xi}, \hat{\mathbf{N}})$ shown in Eq. (4.10).

$(|\mathbf{\Xi}| < \lambda) = \mathbf{0}$. // Constrain the elements in $\mathbf{\Xi}$ whose absolute value smaller than λ as zero during the rest of optimization.

$\hat{\mathbf{X}} = \mathbf{Y} - \hat{\mathbf{N}}$. // Get new estimate of true state.

Calculate $\dot{\hat{\mathbf{X}}}$ using $\hat{\mathbf{X}}$. // Get new estimate of true derivative.

$(|\mathbf{\Xi}| \neq \mathbf{0}) = \mathbf{\Theta}(\hat{\mathbf{X}}) \backslash \dot{\hat{\mathbf{X}}}$. // Regress the dynamics on terms in $\mathbf{\Xi}$ that are not constrained as zero.

B.2 Effect of Thresholding Parameter λ

Thresholding parameter λ is the most important parameter to tune in modified SINDy. The parameter λ will determine the sparsity of the model structure. It's effect can be seen in Fig. B.1. In Fig. B.1, Lorenz equation is simulated with $[-5.0, 5.0, 25.0]$, $dt = 0.01$, and $T = 25$. 10% of Gaussian noise is added and Adam optimizer with learning rate of 0.001 is used to denoise the signal. N_{loop} is set to 8 and different values of λ is used. For each λ, the numerical experiments is performed 10 times to calculate the median and distribution of the error as shown in Fig. B.1. Fig. B.1 suggests that the value of λ must be properly tuned. If the value of λ is too small, the sparsity constraint will not be strong enough to enforce the correct model to be found. Moreover, $\boldsymbol{\Xi}$ and $\hat{\mathbf{N}}$ will easily get stuck in the local minimum. If the value of λ is too large, the correct terms can be wrongly eliminated and the resulting model structure will be wrong. If the model structure is wrong, there will be huge difference between the identified noise $\hat{\mathbf{N}}$ and true noise $\mathbf{N}$. To avoid swiping different values of λ, our proposed method can be easily modified to use the stepwise sparse regression (SSR) approach [200]. However, the use of SSR approach and its performance is not in the scope of this paper.

B.3 Effect of Prediction Step q

Fig. B.2 shows the effect of the prediction step q on the performance of NN denoising approach by Rudy et al. [318] and modified SINDy approach. The chaotic Lorenz system is used for comparison. The Lorenz attractor is simulated by setting $x_0 = [-5, 5, 25]$, $T = 25$, and $dt = 0.01$. The noise level is set to 10% to generate noisy data. Each prediction step is run for 10 times to calculate the median of the error. Adam optimizer, with a learning rate of 0.001 is used to perform the optimization. N_{loop} is set to 3. Fig. B.2 suggests that the performance of modified SINDy is not hugely affected by the prediction step q. However, for the NN denoising approach shown in [318], there exist some value of q to achieve optimal performance. Fig. B.2 also suggests the computational time of both approaches

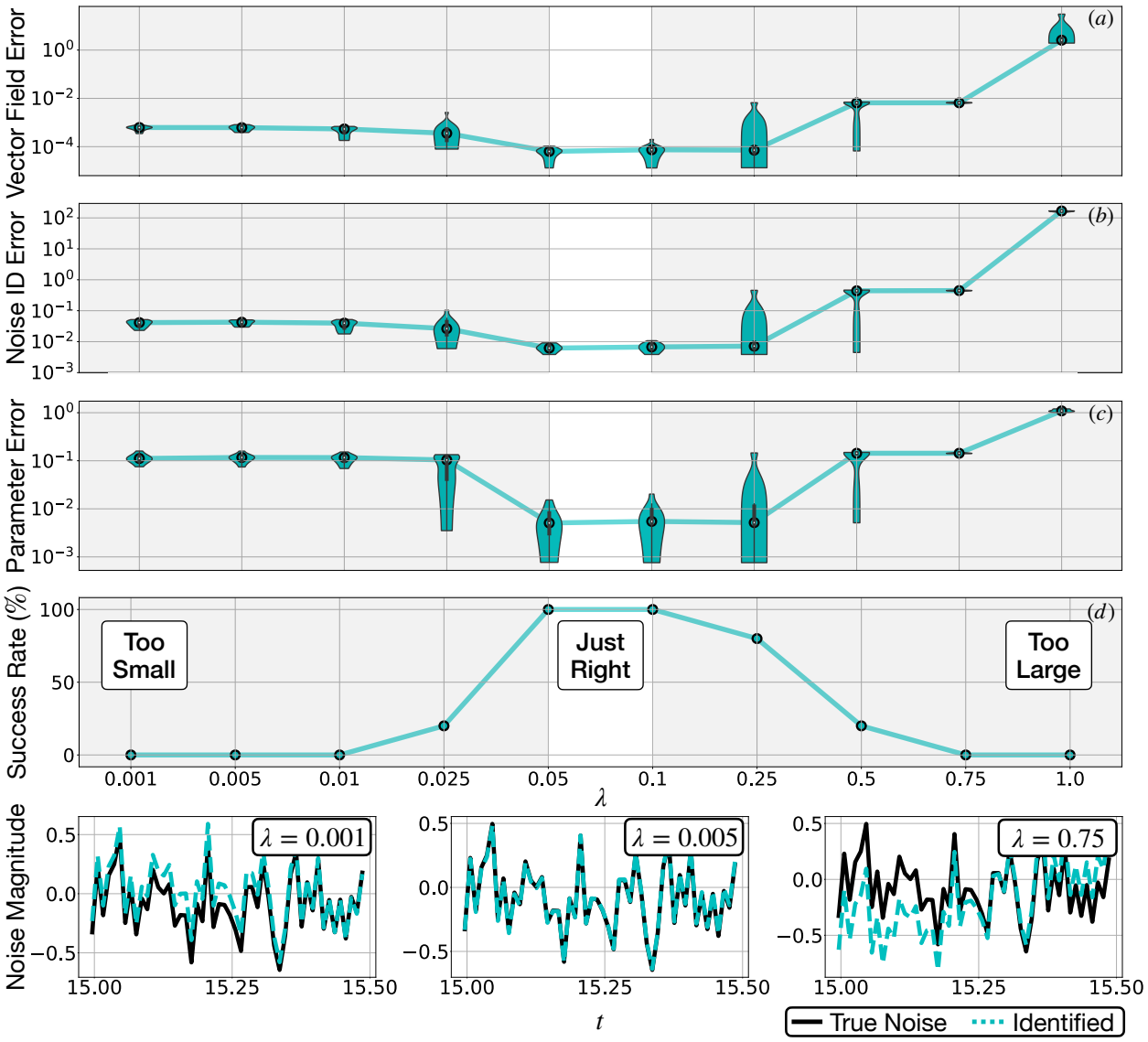


Figure B.1: This figure shows how the choice of sparsity parameter λ will effect modified SINDy performance. If λ is too small, modified SINDy will not converge to the correct model in a short range of time and will be stuck in the local minimum. On the contrary, if the sparsity parameter is too large, the identified model will miss the necessary term to build the correct model. Thus, the value of λ needs to be tuned properly to determine the accurate model.

increase linearly as the value of q increase. Thus, q can be chosen as a small value to save the computational time when using modified SINDy without sacrificing too much of the

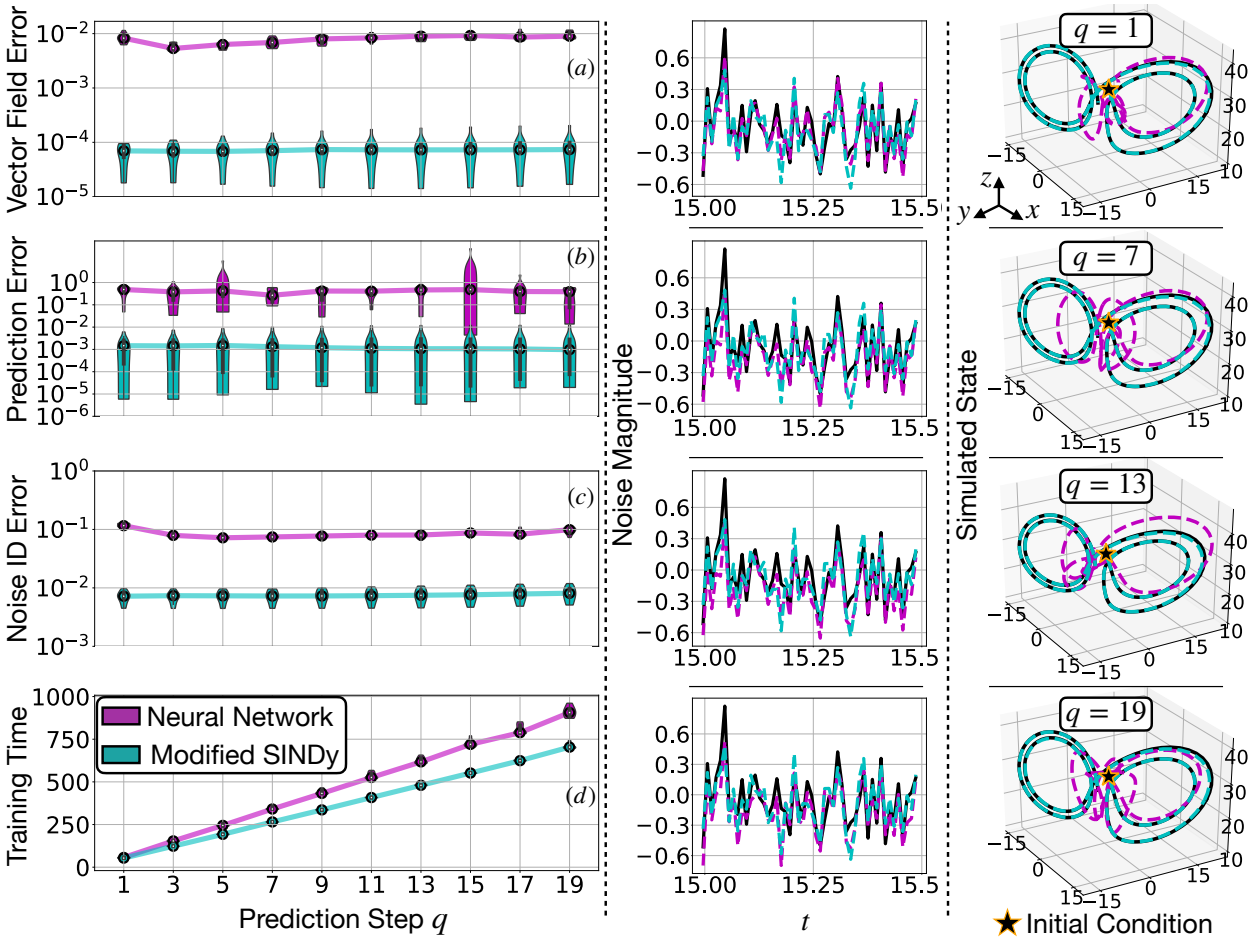


Figure B.2: Left: The value of noise identification error, prediction error, and vector field error of NN denoising approach by Rudy et al. [318] and modified SINDy (labeled as SINDy) are shown as the value of q changes. In (a) to (d), the black circle shows the median of the error of 10 runs while the violin shape represents the distribution of calculated result. Mid: The average of true noise and identified noise of modified SINDy and NN approach is shown for four different prediction steps. Right: The comparison of the simulated and true trajectory of modified SINDy and NN identified model is shown. The simulated trajectory uses $x_0 = [-5, 5, 25]$ and $dt = 0.01$. The model is simulated for 3 seconds. It could be seen that modified SINDy has better performance in this case. All the computation is performed on RTX 2080 GPU, with 32GBs of RAM and AMD Ryzen 7 2700X Processor.

performance.

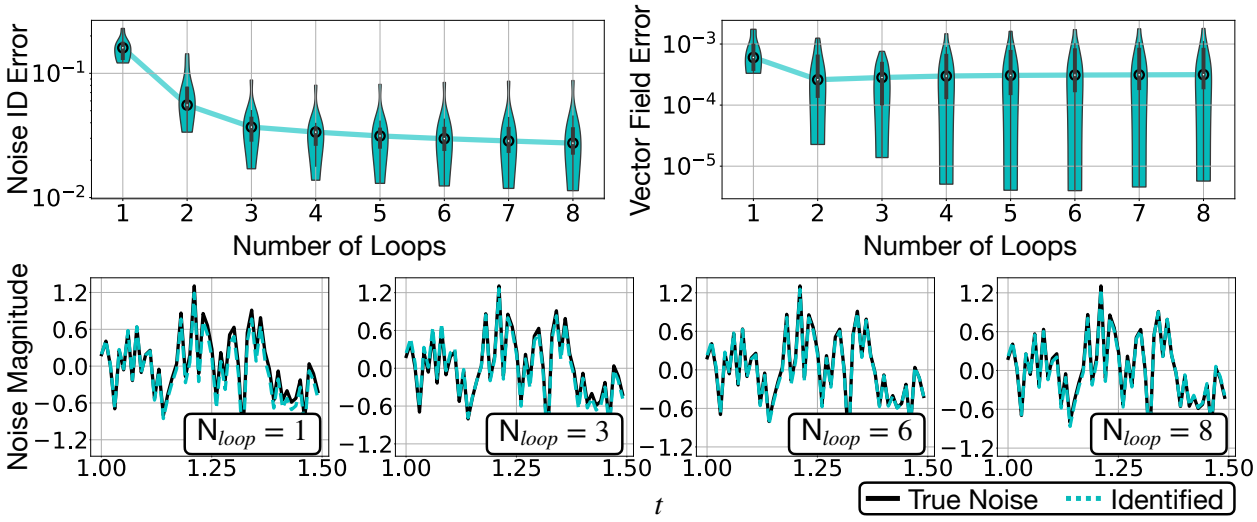


Figure B.3: Top Left: As the number of optimization loop increases, the noise identification error asymptotically decreases and converges to the optimal value. The black circle indicates the median of 10 runs while the violin shape represents error distribution. Top Right: As the number of loop increases, the vector field error gradually converges to a certain value. Bottom: The average of the identified noise and true noise is shown for four different choice of optimization loops. As the number of loop increases, the differences between the true noise and identified noise is minimized.

B.4 Effect of Optimization Iteration N_{loop}

The parameter N_{loop} determines how many times the thresholding optimization is performed. Fig. B.3 shows the effect of N_{loop} on the noise identification error and vector field error using Lorenz attractor as an example. The system is simulated by setting $x_0 = [5, 5, 25]$, $T = 25$, $dt = 0.01$, and $q = 3$. Adam optimizer, with a learning rate of 0.001, is used to optimize the problem. Fig. B.3 suggests the performance of modified SINDy will gradually converge in the end.

B.5 Noise Robustness Comparison with SINDy

This section shows the noise robustness comparison of SINDy [180] and modified SINDy using Van der Pol oscillator, Lorenz attractor, and Rössler attractor. Fig. B.4 shows

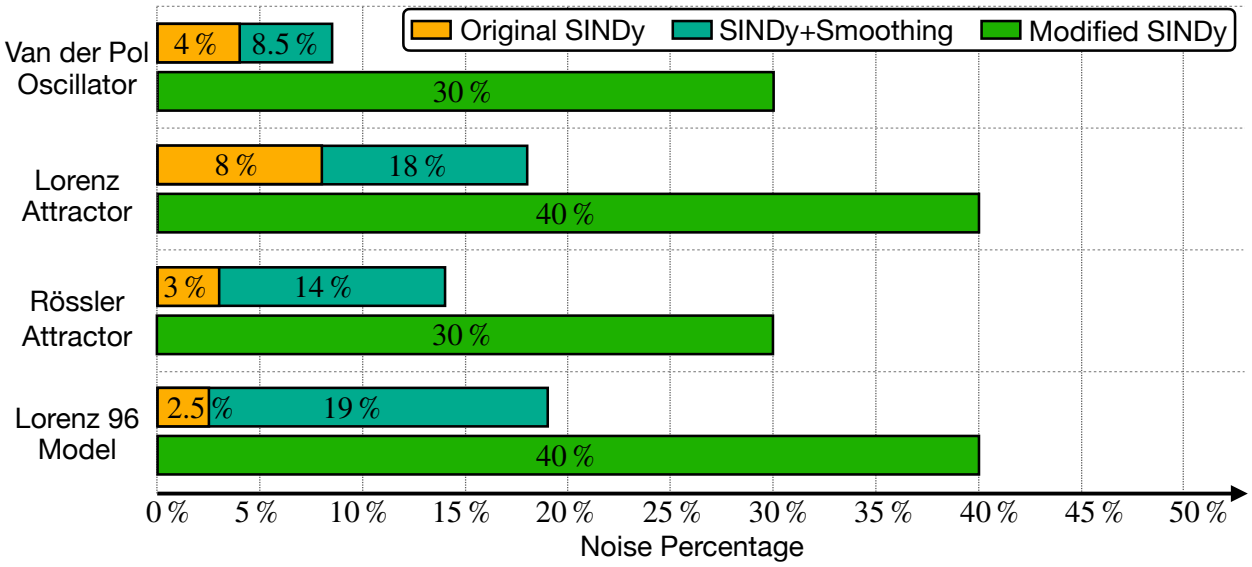


Figure B.4: The maximum level of noise SINDy and modified SINDy can handle to generate the correct model structure is shown. Tikhonov regularization approach is used to pre-smooth the noisy data. It can be seen that the modified SINDy is about 2 times more robust than original SINDy [180].

the maximum noise percentage each algorithm can handle to generate the correct model structure. For each noise level, 5 different noisy data sets are generated and provided to both approaches. If the tested algorithm fails to identify the correct model structure for any noisy data sets at a given noise level, we will assume it is not robust to noise at this level. For SINDy, the derivative is computed using finite difference, and we show the effect of pre-smoothing the noisy data on its performance. Note that no smoothing is applied for modified SINDy. The clean data for Lorenz attractor, Van der Pol oscillator, and Rössler attractor is generated the same way shown in Sec. 4.3.2, Sec. 4.4.1, and Sec. 4.4.2. For SINDy, the sparsity parameter λ is chosen as a hundred uniformly distributed values from 0.01 to the minimum of true parameters' absolute value. For modified SINDy, $q = 1$ and $N_{loop} = 8$ are used for all examples shown in Fig. B.4. Table. B.1 shows other parameters we used for modified SINDy. Note that it is possible to make modified SINDy work at a higher noise level by tuning its parameters. However, swiping various parameters is quite

Table B.1: Parameters used for modified SINDy in Fig. B.4 under maximum noise it can tolerate. The constant term is included when building the library for Rössler attractor and Lorenz 96 model but not for other examples. The parameter error is calculated using Eq. (4.14).

Model	Noise Percentage	Library Order	Random Seed	0	1	2	3	4
Lorenz	30%	2	λ	0.3	0.2	0.3	0.1	0.1
			Parameter Error	0.0046	0.051	0.031	0.032	0.077
			Max Adam Iteration	10000	15000	15000	15000	15000
Rössler	40%	2	λ	0.1	0.22	0.22	0.1	0.1
			Parameter Error	0.021	0.059	0.022	0.0062	0.020
			Max Adam Iteration	15000	15000	15000	15000	15000
Van der Pol	30%	3	λ	0.1	0.22	0.22	0.1	0.1
			Parameter Error	0.053	0.039	0.014	0.011	0.041
			Max Adam Iteration	15000	15000	15000	15000	5000
Lorenz 96	40%	3	λ	0.1	0.15	0.09	0.215	0.1
			Parameter Error	0.015	0.015	0.016	0.034	0.0075
			Max Adam Iteration	7000	7000	10000	10000	5000

computationally heavy for modified SINDy. Thus, the maximum noise level modified SINDy can tolerate in Fig. B.4 is a lower approximate.

B.6 Noise Robustness Comparison with Weak-SINDy

This section shows the noise robustness comparison of Weak-SINDy [233] and modified SINDy using Lorenz attractor as an example. The Lorenz attractor is simulated by setting $x_0 = [5, 5, 25]$, $T = 25$, $dt = 0.01$ and $dt = 0.001$. For both approaches, the library is constructed using up to second order terms (without constant term). Different percentage

of noise is added to the clean data to generate noisy training data. The parameter error and success rate is computed for both approaches. For modified SINDy, Adam optimizer with learning rate of 0.001 is used to perform the optimization. The sparsity parameter is chosen as $\lambda = 0.1$ for most of the time. If the modified SINDy can not produce the correct result, $\lambda = 0.15$ is used instead. When $dt = 0.01$ we pre-smooth the data using approach mentioned in Sec. 4.3.2 and no pre-smoothing is done when $dt = 0.001$. For Weak-SINDy, when $dt = 0.01$, 200 test functions with polynomial order of 14 are used. The width-at-half-max parameter $r_{whm} = 8$, and the support size $s = 31$. When $dt = 0.001$, 1000 test functions with polynomial order of 2 are used. The $r_{whm} = 16$, and $s = 30$. 30 different sparsity parameters evenly ranges from 0 to 0.95 are used, each generates a different candidate model for Weak-SINDy. The final model we used to calculate the parameter error for Weak-SINDy is the model that has correct structure (with only correct terms are selected from the library). If the Weak-SINDy fails to produce the model with correct active terms, the model that predicts the test data best is used to calculate the prediction error, and the test data is generated using initial condition $x_{0,test} = [-10, 10, 15]$ and simulated with $T = 25$ and $dt = 0.01$. The final comparison result of the best model generated by Weak-SINDy and modified SINDy can be seen in Fig. B.5. Although Fig. B.5 suggests the modified SINDy has better performance in high noise scenarios, it does have higher computational cost. When using a second order library and the above mentioned parameters, the Weak SINDy takes 0.073 second to compute the selection matrix $\hat{\boldsymbol{\Xi}}$ for a given sparsity parameters λ. On the other hand, modified SINDy takes around 35 seconds to compute $\hat{\boldsymbol{\Xi}}$ for a given sparsity parameters while $N_{loop} = 3$ and $q = 1$. Thus, Weak SINDy is two orders of magnitude faster than modified SINDy. Compared to backslash operation (Matlab's **mldivide**) used in Weak SINDy, the ADAM optimizer used in modified SINDy is slow. Our future work includes speeding up the modified SINDy calculation speed.

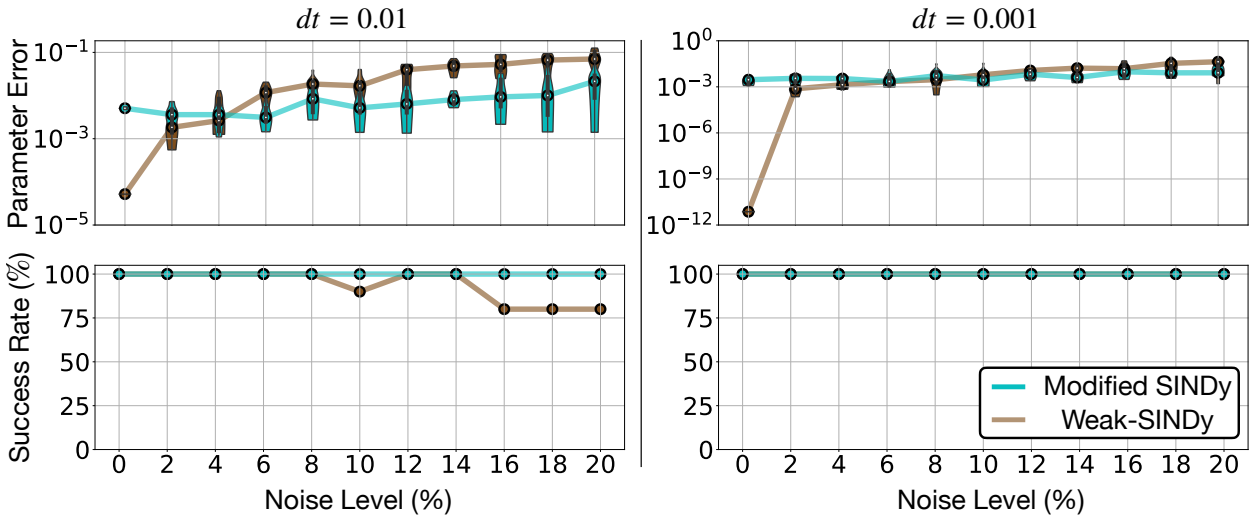


Figure B.5: This figure shows noise robustness comparison of modified SINDy and Weak-SINDy using Lorenz attractor as an example. The effect of noise on the parameter error and successful identification rate is compared for both approaches. As it shows in the figure, Weak-SINDy and modified SINDy has almost the same accuracy when $dt = 0.001$. However, when $dt = 0.01$, the modified SINDy has a slightly higher success identification rate.

Table B.2: Parameters used in Fig. 4.8

Models	Initial Condition	Library Order	Learning Rate	T	q	λ
Van der Pol	$[-2, 1]$	3	0.001	10	1	0.05
Duffing	$[-2, -2]$	3	0.001	25	1	0.05
Cubic	$[0, 2]$	3	0.001	25	1	0.08
Lotka-Volterra	$[1, 2]$	3	0.001	10	1	0.2
Lorenz	$[5, 5, 25]$	2	0.001	25	3	0.1,0.15

B.7 Parameters Used in Fig. 4.8

In this section, the models used to simulate the system in Fig. 4.8 are listed. The model used for simulating the Duffing oscillator is

$$\begin{aligned} \dot{x} &= y, \\ \dot{y} &= -p_1 y - p_2 x - p_3 x^3, \end{aligned} \tag{B.1}$$

with $p_1 = 0.2$, $p_2 = 0.1$, and $p_3 = 1$. The model used for simulating the Cubic oscillator is

$$\begin{aligned} \dot{x} &= p_1 x^3 + p_2 y^3, \\ \dot{y} &= p_3 x^3 + p_4 y^3, \end{aligned} \tag{B.2}$$

with $p_1 = -0.1$, $p_2 = 2$, $p_3 = -2$, and $p_4 = 0.1$. The model used for simulating the Lotka-Volterra system is

$$\begin{aligned} \dot{x} &= p_1 x - p_2 xy, \\ \dot{y} &= p_2 xy - 2p_1 y, \end{aligned} \tag{B.3}$$

with $p_1 = 1$ and $p_2 = 0.5$. Other parameters used for training the modified SINDy is summarized in Table. B.2. For all examples, $N_{loop} = 5$ and $dt = 0.01$.

B.8 Tips on Learning Non-Zero Mean Noise

As Sec. 4.4.4 suggests, learning non-zero mean noise distribution is much harder than learning the zero-mean noise distribution. To achieve better performance on the non-zero mean noise distribution, we propose an iterative learning approach. This approach can be summarized as follow: 1. Apply modified SINDy to the noisy data, briefly learn the distribution of noise. 2. Subtract the mean of learned noise from the noisy measurement, and use the new data to perform the learning. 3. Repeat the step 2 until the result converges and the correct model is found. The Van der Pol oscillator is used to illustrate this approach, and the clean data is generated the same way in Sec. 4.4.1. 20% of Gamma noise is added to

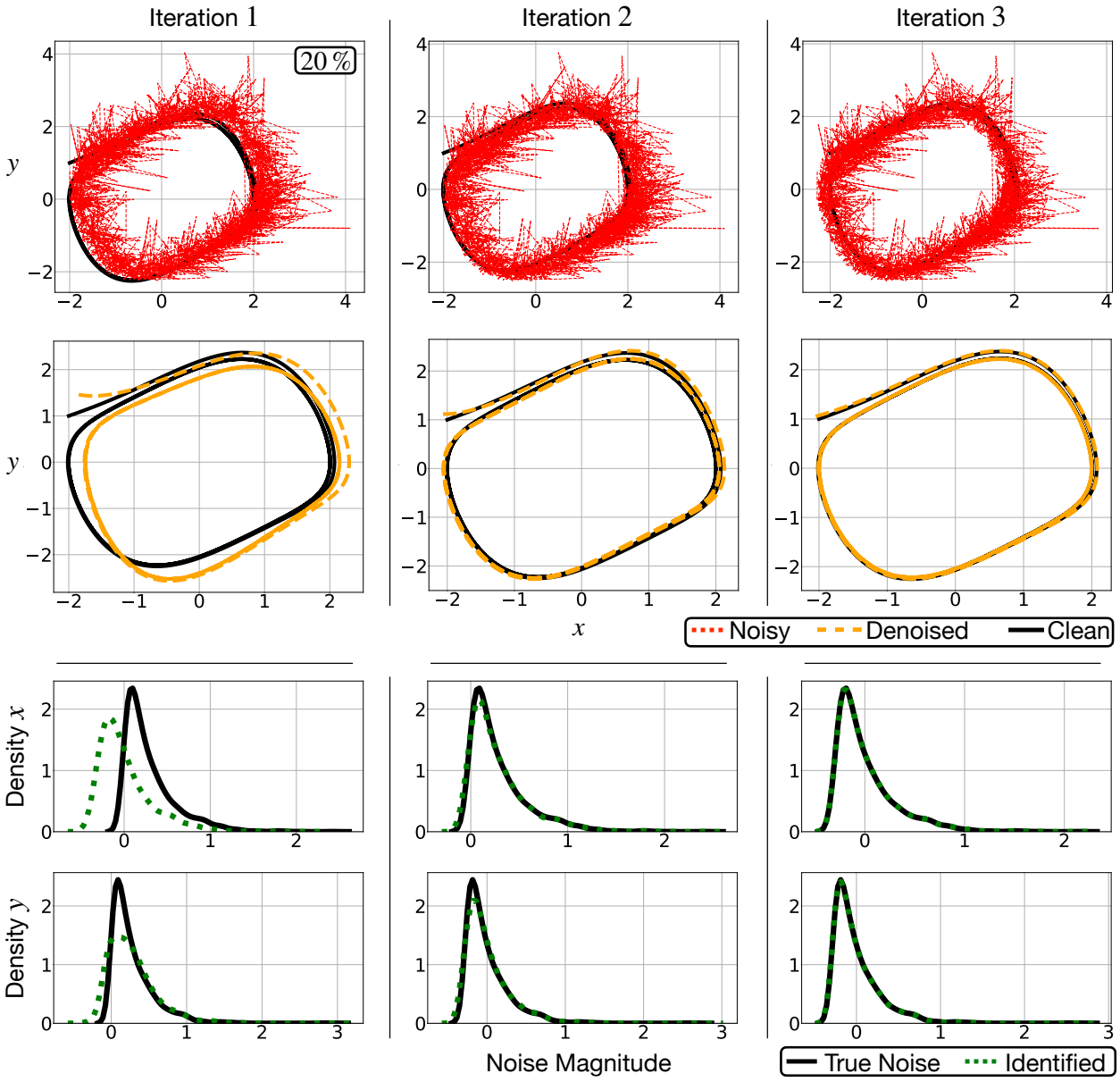


Figure B.6: Iterative learning process is shown to tackle the non-zero mean noise distribution. By using this iterative process, modified SINDy can tackle noise distribution with non-zero means.

create the noisy data. The parameters are set as $q = 2$, and $\lambda = 0.15$. Fig. B.6 demonstrate this approach. However, there's no guarantee that this approach will work when the bias

Table B.3: The identified noise distribution versus the true distribution. For Gamma distribution, the parameter k represents shape and θ represents the scale.

True Distribution	State	True Parameter	Identified Distribution	Identified Parameters
Gaussian	x	$\mu = 0, \sigma = 0.1413$	Gaussian	$\hat{\mu} = 0.003, \hat{\sigma} = 0.1451$
	y	$\mu = 0, \sigma = 0.1439$	Gaussian	$\hat{\mu} = 0.009, \hat{\sigma} = 0.1439$
Uniform	x	$\mu = 0, \sigma = 0.1413$	Uniform	$\hat{\mu} = -0.0717, \hat{\sigma} = 0.1438$
	y	$\mu = 0, \sigma = 0.1439$	Uniform	$\hat{\mu} = -0.0729, \hat{\sigma} = 0.1466$
Gamma	x	$k = 1, \text{loc} = 0, \theta = 0.1413$ $\mu = 0.1413, \sigma = 0.02$	Gamma	$k = 3.2714, \text{loc} = -0.095, \theta = 0.0722$ $\hat{\mu} = 0.1409, \hat{\sigma} = 0.0211$
	y	$k = 1, \text{loc} = 0, \theta = 0.1439$ $\mu = 0.1439, \sigma = 0.021$	Gamma	$k = 10.49, \text{loc} = -0.3105, \theta = 0.0432$ $\hat{\mu} = 0.1419, \hat{\sigma} = 0.0217$
Dweibull	x	$c = 2.07, \text{loc} = 0,$ scale $= 0.1413$	Dweibull	$\hat{c} = 2.064, \text{loc} = 0.8 \times 10^{-5},$ scale $= 0.1408$
	y	$c = 2.07, \text{loc} = 0,$ scale $= 0.1439$	Dweibull	$\hat{c} = 2.048, \text{loc} = -2.8 \times 10^{-5},$ scale $= 0.1438$
Rayleigh	x	$\mu = 0.1775, \sigma = 0.0085$	Rayleigh	$\hat{\mu} = 0.1775, \hat{\sigma} = 0.0085$
	y	$\mu = 0.1779, \sigma = 0.0086$	Rayleigh	$\hat{\mu} = 0.1779, \hat{\sigma} = 0.0086$

of noise is too large, learning the non-zero mean noise is quite hard and careful tuning is needed. We find out using the soft start approach will also help the denoising of non-zero mean noise.

B.9 Identifying Noise Distribution Type

When the noise is identified, it might be interesting to learn what type of distribution the noise follows. To illustrate this, the Van der Pol oscillator shown in Eq. (4.17) is simulated with initial condition $[-2, 1]$, $T = 50$, and $dt = 0.001$ (for Gamma and Rayleigh noise distribution, $dt = 0.01$). Next, 10% of noise is added to the simulation data to generate the noisy data. The noisy data is provided to modified SINDy to learn the dynamics and identify the noise added to the signal. We set $q = 2$, $\lambda = 0.15$ ($\lambda = 0.2$ for Gamma noise). Adam optimizer with learning rate equals to 0.001 is used, and the library order is set to 3. For all cases, the modified SINDy correctly identified the model. As Table. B.3 shows, five different noise distributions is used to generate the noisy data. After the noise is identified,

the distribution of noise is fitted into seven candidate noise distributions, which are normal distribution, uniform distribution, Gamma distribution, Dweibull distribution, Rayleigh distribution, Cauchy distribution, and Beta distribution. Next, the sum of the square errors between the $\hat{\mathbf{N}}$ and the fitted distribution is calculated, and the distribution that produces the lowest error is selected as the identified noise distribution. Notice that when there's not enough data provided, it is totally possible that other kinds of distribution is misidentified as the true underlying distribution of noise. The study of how many data points is needed to identify the correct distribution is beyond the scope of this paper. The final result can be summarized in Table. B.3.

B.10 Caveats of the Approach

This section provides some tips on using modified SINDy.

1. Properly design the library: Building the correct library for the regression is the most important part of this algorithm. If the library does not contain the terms included in the actual dynamics, the algorithm will fail to produce the correct noise and system model. Thus, whenever possible, one should include any prior information of the dynamics to build the library. In general, the library needs to be large enough to include all the terms that show up in the dynamics, and at the same time small enough to ensure the robustness. The best way to design the library is an open problem in the SINDy framework. When designing the library, one should use as much expert knowledge as possible. Usually, when some expert knowledge of the system is known, it can help user avoid unnecessary higher order terms. For example, when the signal is obtained from planetary system, then one might set the highest order of the library to 3, since cubic term shows up in the Newton's universal law of gravitation. When this kind of expert knowledge is known, usually physical law governing the dynamic, it can give us some idea on how to pick the highest polynomial term. When there's no expert knowledge available, we advise the reader

start with simple second order library and see how the identified model performs. If the performance of the model is not ideal, then use one order higher library until the well performed model is identified. However, keep in mind that using a really high order library will make the optimization numerically sensitive, and at the same time increases computational burden, as Fig. B.7 shows. Do not expect the modified SINDy will work on a library with hundreds or thousands of terms, it will break if the library is too large. For example, when using the Lorenz example with above 20% noise, the maximum order of the library modified SINDy can handle is 4 (about 34 terms). This happens since the higher order terms in the library will tend to mess up the forward and backward simulation and producing the `nan` cost, making the optimizer fails. To leverage this, one can try to decrease the learning rate of the optimizer, pre-smooth the data, get better initial estimate of $\boldsymbol{\Xi}$, reduce the library size, or set optimization parameters type as `float64`. Moreover, whether the constant term $\mathbf{1}$ should be included is case-specific. If the actual dynamics do not have a constant term and the measurement noise is non-zero mean or has significant outliers, including the constant basis in the library will make modified SINDy get stuck at the local minimum more easily. It is advised that the user tries both the library with and without constant basis.

2. Initial guess of $\hat{\mathbf{N}}$ and $\boldsymbol{\Xi}$: Having a good initial guess of the estimated noise $\hat{\mathbf{N}}$ and estimated selection parameter $\boldsymbol{\Xi}$ can improve the condition of the optimization problem and allowing us tackle harder problem with more library terms. If possible, the initial values of $\hat{\mathbf{N}}$ can be obtained by pre-smoothing the noisy signal, which will provide a good start for the optimization problem, and it is also good for estimating $\boldsymbol{\Xi}$. If no other information is given, the initial guess of the $\hat{\mathbf{N}}$ can be set as zeros.

3. The choice of N_{train} and N_{loop}: The proper choice of parameter N_{train} and N_{loop} is also important for the successful identification of the system. The parameter N_{train} determines how many times the gradient descent step is applied. It also determines

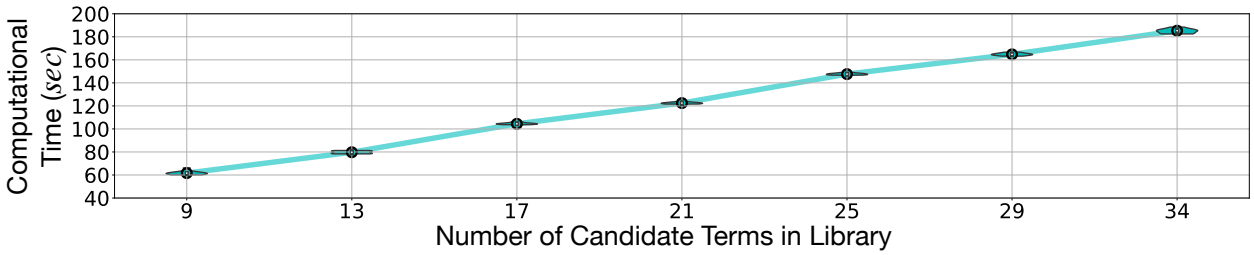


Figure B.7: This figures shows the computational time needed to run the ADAM optimizer for 5000 iterations with different numbers of candidate functions. The Lorenz attractor is used as an example, with $T = 25$, $dt = 0.01$, and noise level set to 20%. This computation is performed 8 times for each number of candidate functions. When the number of candidate function is 9, it corresponds to second order library, and when the number of candidate function is 34, it corresponds to fourth order library. As it shows, the computational time increases almost linearly with the number of candidate terms in the library.

after how many gradient descent steps should the sparsity constraint be applied to $\boldsymbol{\Xi}$. For the examples used in this paper, a good choice is $N_{train} = 5000$, when the learning of ADAM is set to 0.001. As for N_{loop}, it should be a sufficiently large number that make sure the final answer converges. As seen in Fig. B.3, after 8 iteration of loops, the noise identification error and vector filed error converged. This indicates a good choice of N_{loop} is the one that drives the noise identification error or vector filed error below certain user define threshold or makes the error indicator converge. Moreover, when the noise level is high, N_{loop} should also be set higher. For all the examples used in this paper, $N_{loop} = 8$ is a good choice.

B.11 Global Minimum of Cost Function Eq. (4.9)

The cost function shown in Eq. (4.9) has multiple global optimum solutions capable of achieving the optimum cost $L = 0$. Moreover, as the dimensionality of the free parameters $\hat{\mathbf{N}}$ and $\boldsymbol{\Xi}$ exceeds the dimensionality of the data $\mathbf{Y}$, the optimization problem is ill-posed. Here, we summarize several global optimal solutions of Eq. (4.9):

- The global optimum can show up if we can pick $\hat{\mathbf{N}} + \mathbf{C} = \mathbf{Y}$, which will result

$\hat{\mathbf{X}} = \mathbf{Y} - \hat{\mathbf{N}} = \mathbf{C}$, where $\mathbf{C}$ is a constant. Thus, when picking $\boldsymbol{\Xi} = \mathbf{0}$, we can obtain a trivial solution that achieve the global optimum. In this case, the structure of the identified model is the sparsest. To completely avoid this solution, one can add additional penalty on the noise, such that it penalize $\hat{\mathbf{N}}$ to have large magnitude and avoid $\hat{\mathbf{N}} = \mathbf{Y}$. However, in practice we end up not adding this penalty term in our final optimization problem. We did this for several reasons:

1. It help us avoid adding additional tuning parameters which will make choosing the correct hyper parameter difficult.
2. Usually, the thresholding parameter λ is picked so that it is smaller than the maximum value of Ξ. Thus, it is unlikely to generate $\hat{\boldsymbol{\Xi}} = \mathbf{0}$.
3. Moreover, the initial guess of $\hat{\mathbf{N}}$ is picked as zero or other values that is close to true $\mathbf{N}$ (when using warm start). Thus, this initial guess of $\hat{\mathbf{N}}$ is quite small compared to the magnitude of $\mathbf{Y}$, which means sufficiently large iterations is needed to achieve $\hat{\mathbf{N}} = \mathbf{Y}$ when using ADAM optimizer. This indicates it is more natural for $\hat{\mathbf{N}}$ converge to $\mathbf{N}$ rather than $\mathbf{Y}$.

We recreated this situation in Figure. B.8. To generate this global optimum, we used Lorenz system with second order Polynomial library. The noisy data set is generated by using initial condition $x_0 = [5, 5, 25]$, $dt = 0.01$, $T = 25$, and 20% of Gaussian noise (using random seed 4). We picked our thresholding parameter $\lambda = 20$ so that we enforces $\hat{\boldsymbol{\Xi}} = \mathbf{0}$. Then we used ADAM optimizer with learning rate 0.001 to solve the optimization problem with $N_{loop} = 16$ and $N_{iter} = 5000$. As Fig. B.8 shows, when enforcing $\hat{\boldsymbol{\Xi}} = \mathbf{0}$, we have $\hat{\mathbf{N}} = \mathbf{Y} + \mathbf{C}$. Note, to generate this result, we did not add penalty term on the magnitude of noise.

- The second global optimum is the true system we wish to identify. For example, when $\hat{\mathbf{N}} = \mathbf{N}$, we have $\hat{\mathbf{X}} = \mathbf{Y} - \hat{\mathbf{N}} = \mathbf{X}$. Thus, if we can pick the correct selection

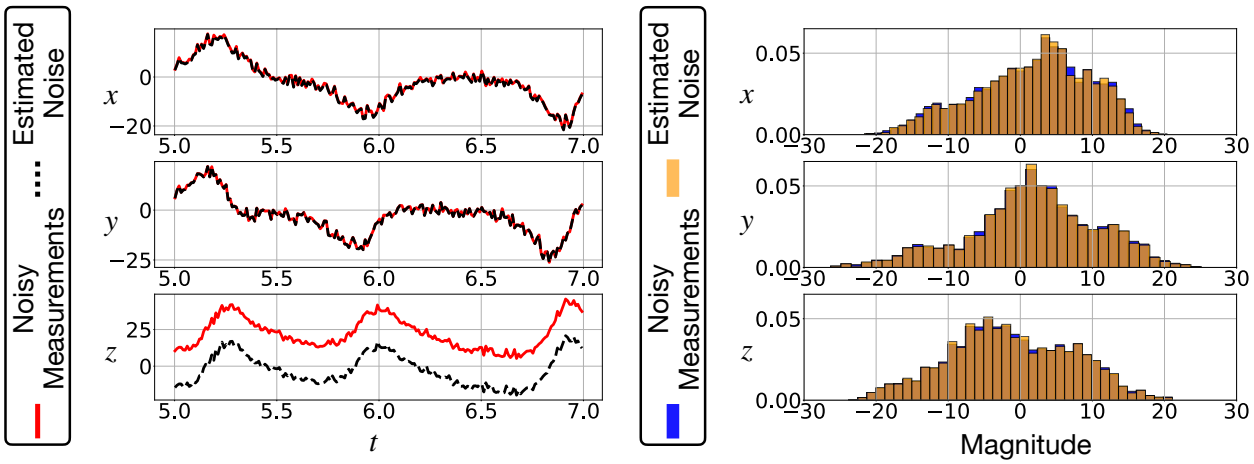


Figure B.8: This figures illustrates one of the global minimum solution to Eq. (4.9). As we can see, the estimated noise $\hat{\mathbf{N}}$ differs from the noisy measurement $\mathbf{Y}$ by a constant value $\mathbf{C}$. This solution along with $\hat{\mathbf{\Xi}}$ forms one of the global optimum of cost function Eq. (4.9). This solution is unlikely to happen as long as $\hat{\mathbf{\Xi}}$ is not zero or an additional penalty term on $\hat{\mathbf{N}}$ is added to the cost function Eq. (4.9).

vector $\hat{\mathbf{\Xi}} = \mathbf{\Xi}$, we will achieve zero cost value. This solution is what we are looking for.

- The third case is more interesting, and we find out it tends to happen when the noise added to the system is extremely high (for example 40% to 100% of noise). In this case, our final solution will tend to converge to a "sister system" that has similar behavior to the actual system.

  For example, suppose we are using a polynomial library, and we wish to identify the Lorenz system given its noisy simulation data with 50% Gaussian noise added. The noisy data set is generated by using initial condition $x_0 = [5, 5, 25]$, $dt = 0.01$, $T = 25$, and the Gaussian noise is generated using random seed 5. We picked our thresholding parameter $\lambda = 0.1$ and ADAM optimizer with learning rate 0.001 to solve the optimization problem with $N_{loop} = 8$ and $N_{iter} = 5000$. As Fig. B.9 shows, then final denoised system converged to a "sister system" that has similar behavior to

the original Lorenz system, with final cost value 0.001679. In this case, the identified system is

$$\begin{aligned} \dot{x} &= -1.26x + 7.87y - 0.243xz, \\ \dot{y} &= 24.33x - 0.951xz + 0.787y, \\ \dot{z} &= 0.932xy - 2.58z, \end{aligned} \tag{B.4}$$

with $E_{\mathbf{N}} = 1.15$, $E_{\boldsymbol{f}} = 0.0186$, and $E_{\mathbf{F}} = 0.436$. This example suggests that it is possible to find out a $\hat{\boldsymbol{\Xi}} \neq \boldsymbol{\Xi}$ and $\hat{\mathbf{N}} \neq \mathbf{N}$, such that the final cost value is still close to zero, which suggests the optimization problem in Eq. (4.9) without regularization term can theoretically have infinite global optimum. In other words, for any constant value of $\hat{\boldsymbol{\Xi}}$, there exist a unique $\hat{\mathbf{N}}$ that makes the cost function in Eq. (4.9) equals to zero.

In practice, we find out the third global optimum tends to show up when the noise-signal ratio is high (usually above 40%). Moreover, in this case, the identified noise tends to have a slightly larger magnitude than the true noise, as Fig. B.9 illustrates. Thus, when dealing with highly corrupted measurement data, it is recommended the reader add a L_2 regularization term on the estimated noise magnitude to avoid the this type of global optimum. When the noise-signal ratio is low (under 30% for all the example we used in this paper), we find out this type of solution does not show up as long as the correct thresholding parameter is picked, we believe this is due to the initial guess of $\hat{\boldsymbol{\Xi}}$ and $\hat{\mathbf{N}}$ is good enough to avoid this type of solution.

To summarize, the optimization problem shown in Eq. (4.9) has multiple global optimum solution. When the noise-signal ratio is low and the initial guess of $\hat{\boldsymbol{\Xi}}$ and $\hat{\mathbf{N}}$ is good enough and proper thresholding parameter λ is picked, the final solution will converge to the true dynamics and noise. Moreover, it is recommended to add noise regularization term when the added noise is too high. The effect of additional regularization term on the performance of approach, and the proper way to tune this regularization term is beyond the scope of this paper.

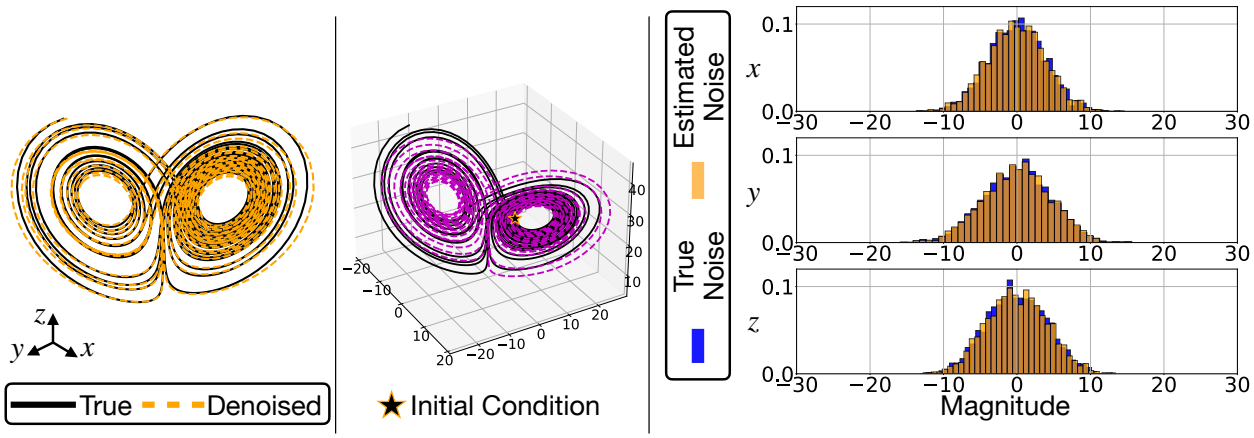


Figure B.9: This figures illustrates the third type of global minimum solution to Eq. (4.9). Left: Even though the denoised signal does not perfectly match the underlying Lorenz system, its overall behavior resembles it. Middle: The simulation result of identified system shown in Eq. (B.4). Even though there's model discrepancy between the identified system and true system, the identified system still captures the shape of the Lorenz attractor. The identified system is simulated using initial condition $x_0 = [5, 5, 25]$, with $dt = 0.01$ and $T = 25$. Right: The distribution of identified noise and true noise. It can be seen that the identified noise has larger magnitude than the ground truth, which suggests adding a noise magnitude regularization term can help us converge to the true system.

B.12 Calculation of the derivative term e_d

The derivative error shown in Eq. (4.3) and Eq. (4.9) are calculated using five-point stencil approach throughout the paper. Given an estimated state vector $\hat{\mathbf{X}} \in \mathbb{R}^{m \times n}$, the derivative is calculated as

$$\dot{\hat{\mathbf{X}}} = \frac{-\hat{\mathbf{X}}[5 : end, :] + 8\hat{\mathbf{X}}[4 : end - 1, :] - 8\hat{\mathbf{X}}[2 : end - 3, :] + \hat{\mathbf{X}}[1 : end - 4, :]}{12dt}, \tag{B.5}$$

where $[i : j, :]$ represent the matrix slicing. The resulting derivative estimation term will have shape $\dot{\hat{\mathbf{X}}} \in \mathbb{R}^{m-4 \times n}$. Thus to calculate the derivative error e_d, we discard the first and last two rows of matrix $\hat{\mathbf{X}}$ so that the dimension of $\dot{\hat{\mathbf{X}}}$ match with $\boldsymbol{\Theta}(\hat{\mathbf{X}}[3 : end - 2, :])\boldsymbol{\Xi} \in$

Table B.4: This table shows the denosing performance difference of using Eq. (4.9) and using Eq. (4.9) without derivative term e_d on Lorenz example with 40% of noise.

Terms	Cost Functrion: Eq. (4.9)	Cost Function: Eq. (4.9) without e_d
$E_{\boldsymbol{f}}$	0.00259	0.0011
$E_{\mathbf{N}}$	0.271	0.1975
$E_{\mathbf{F}}$	0.003	0.00136
$E_{\boldsymbol{p}}$	0.01259	0.0158
$\dot{x}$	$-9.7206x + 9.796y$	$-9.673x + 9.759y$
$\dot{y}$	$27.828x - 1.032y - 0.988xz$	$28.275x - 1.089y - 1.003xz$
$\dot{z}$	$-2.729z + 1.059xy$	$-2.678z + 1.033xy$

$\mathbb{R}^{m-4\times n}$.

It is also worth noting that the simulation error e_s already constraints the derivative error e_d. In other words, when minimizing the simulation error e_s, the derivative error is also minimized, since e_s utilize the right-hand side of system dynamics $\boldsymbol{\Theta}(\hat{\mathbf{X}})\boldsymbol{\Xi}$. Thus, we find out dropping the derivative error term e_d from the cost function Eq. (4.9) does not affect the overall performance of the approach. To illustrate this, the Lorenz attractor is simulated using $dt = 0.01, T = 25$ with initial condition $x_0 = [5, 5, 25]$. Then 40% of noise is added to the simulation data to generate noisy measurement. Next, ADAM optimizer is used with following parameters to optimize the cost function shown in Eq. (4.9). The parameters used is $q = 1$, $N_{train} = 5000$, $N_{loop} = 8$, learning rate is set to 0.001, and the order of library is 2. Next, the same parameters is used to minimize the cost function without the derivative error term. The final result yield by using two cost function can be seen in Table. B.4. Note that $E_{\mathbf{F}}$ is calculate by simulating the identified system using initial condition $x_0 = [5, 5, 25]$, $dt = 0.01$ and $T = 3$. As Table. B.4 suggests, dropping the e_d term from the cost function Eq. (4.9) does not change the final result too much. Moreover, the $E_{\boldsymbol{f}}$, $E_{\mathbf{N}}$, and $E_{\mathbf{F}}$ error got slight improvement. We believe this happens since dropping the e_d term from the cost function avoids the numeric derivative approximation error.

Appendix C

APPENDIX FOR CHAPTER 6

C.1 Real-Time System Using Simulink Desktop Real-Time

Besides using the Speedgoat Real-Time machine and Simulink Real-Time, it is possible to use a low-cost alternative as the Real-Time system. This solution uses the National Instrument Data Aqcusition (DAQ) device PCIe-6341 as the data collection device (I/O module). PCIe-6341 can read up to four quadrature encoders at the same time. Moreover, it has multiple digital I/Os and analog I/Os. For more details on the device specification, please check the user manual. In this setup, the sensors are connected to the National Instrument terminal connector SCB-68A. Then, the terminal connector and DAQ are connected using cable SHC68-68-EPM. This connection completes the sensor data acquisition task. To read sensor signals in real-time and send control command (analog signal) to the motor drive (velocity mode), the Simulink Desktop Real-Time is used as the real-time operating system. In this case, only one computer is needed to develop the user program and perform real-time control. This real-time system setup has been tested in our previous paper [178]. When using Simulink Desktop Real-Time, the real-time program only runs on a single core instead of multi-cores as Simulink Real-Time does. This drawback decreases the maximum sampling frequency of the Simulink Desktop Real-Time setup. Our previous experiments show the maximum sampling rate in pure data-collection mode is around $5kHz$ when using Simulink Desktop Real-Time with the host machine we mentioned in Sec. 6.5. The maximum sampling rate of the double pendulum stabilization task (using time-varying LQR and Kalman filter) is around $2kHz$ in this case.

C.2 Pendulum Arm Details

C.2.1 Design Details

An overview of our pendulum arm's design is described in section 6.3. Here, we describe the design of the pendulum arm in more detail. The main components of the pendulum arm are: 1) pendulum body, 2) shaft, 3) bearing plate, and 4) 3D printed protection case. The overall structure of the pendulum arm is determined by how it transmits the rotational information measured by the encoders. In our design, a slip-ring sends the encoders' electrical signals (mainly the A-, B-, and I-channel signals) to the real-time system, and conducts currents to the encoder sensors on the rotational component. The slip-ring design introduces friction on the contact between the slip-ring shaft and brush block. To minimize this additional effect of friction, a miniature slip-ring and slip-ring brush are used with gold contact surfaces that reduce the electrical noise during the rotational movement of the pendulum arm. Other designs based on vision tracking or wireless technology to measure the pendulum arm rotational angle could avoid these additional friction effects. However, the slip-ring design is simple and does not have latency in the signal transmission that occur in vision-based and wireless communication type pendulum arms. Also, no additional computational resources are needed to determine the rotational angle of the pendulum compared to vision-based tracking systems. This characteristic is particularly beneficial for achieving higher sampling rates. The challenge of using a slip-ring is that it requires precision machining of the pendulum shaft. Moreover, the slip-ring has a fixed number of channels to transmit signals. This may reduce the flexibility of the setup if new sensors (e.g. an inertial measurement unit (IMU) sensor) are required for future experiments. To accommodate the slip-ring and to connect it with the sensors, the first and second pendulum arm's shaft is designed to be hollow, as shown in Fig. C.1. As the first pendulum arm shaft takes most of the static and dynamic load, it has a larger diameter than the second and third pendulum arm shafts. Moreover, to minimize the axial oscillation of the pendulum arm, a thread is machined on the first pendulum

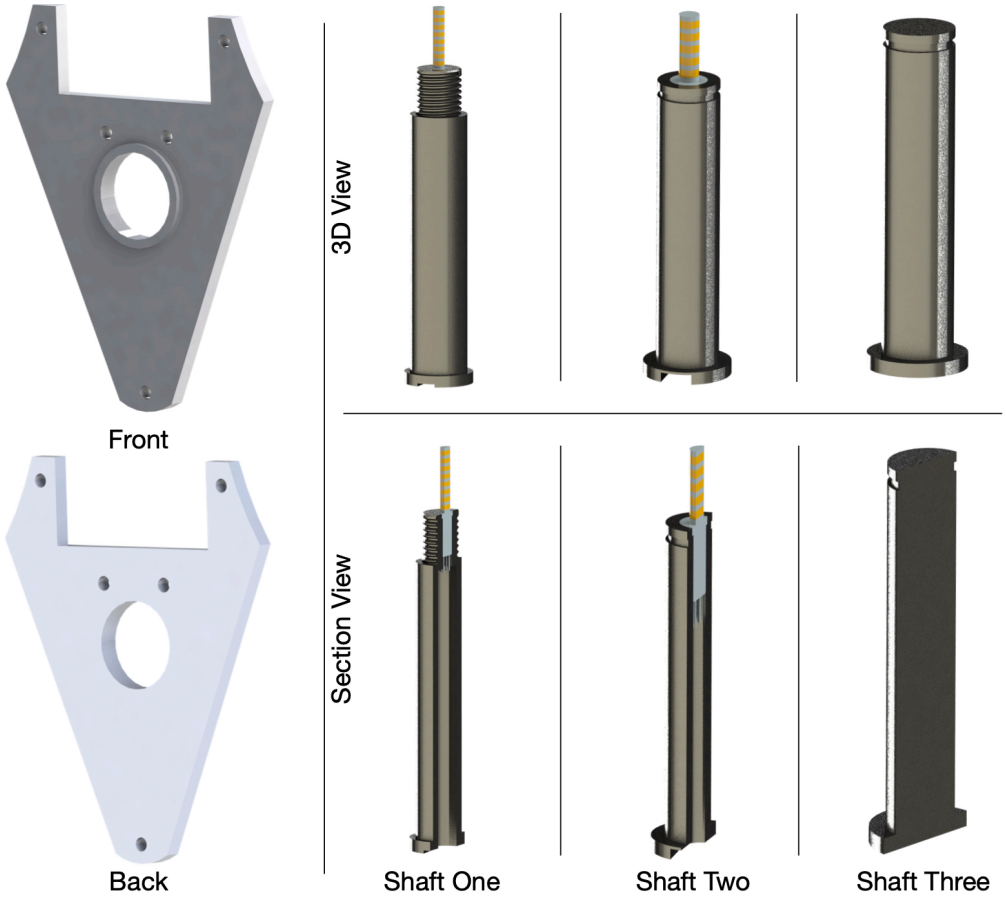


Figure C.1: Design of the bearing plate and pendulum arm's shaft. The bearing can be installed on the bearing plate and together support the free rotational movement of the pendulum shaft and pendulum arm. The pendulum arm's shaft is designed hollow to accommodate the slip-ring wires. The groove on the bottom of the pendulum shaft is used to place guide the slip-ring wires into the pendulum arm.

shaft such that the axial load of the pendulum arm is adjustable. When more axial force is applied, we can reduce the axial oscillation of the pendulum arm.

The slip-ring wires enter the pendulum arm through a small hole on the front side of the pendulum arm, near the position where the pendulum shaft is mounted, as shown in Fig. C.2 (A). The slip-ring wire enters the pendulum arm through this hole and connects with the encoder located on the backside of the pendulum arm. The groove in Fig. C.2 (A) is aligned with the groove on the pendulum arm's shaft, as shown in Fig. C.1. Together,

they guide the slip-ring wire and provide space to store it and avoid the cable to stick out of the front face of the pendulum arm. To properly manage the slip-ring wires inside the pendulum arm, a 3D printed cable clip is designed, shown in Fig. 6.4 (B). This cable clip holds the slip-ring wire to its place and prevents twining of the slip-ring wire. To reduce the weight of the pendulum arm, several holes are drilled near the shaft hole, indicated in Fig. C.2 (B). Moreover, two holes on the side of the pendulum arm facilitate the installing of the encoder disk on the pendulum shaft, shown in Fig. C.2 (C). In Fig. C.2 (C), the stair case shoulder is shown that properly secures the bearing that is installed on the pendulum arm. This stair case shoulder is not needed in the third pendulum arm since no other pendulum arm is attach to it, shown in Fig. C.2 (D).

Ceramic bearings are used to minimize the friction during the rotational movement. They are installed on the bearing plate shown in Fig. C.1 and the pendulum arm shown in Fig. C.2. The great advantage of the ceramic bearing is that it operates without lubrication, effectively decreasing the drag force caused by lubricants. Moreover, a bearing without sealing is used to further reduce the friction force, since the pendulum setup is used in a clean lab space. Thus, special care must be taken when assembling and dissembling the pendulum arm to ensure that no dust or chips enter the bearing during the process. Two bearings are used to fully support the rotational movement of the pendulum arm. Moreover, the bearings are aligned by connecting the bearing plate and pendulum arm body together. Together, they form a bearing housing where the shaft is installed. To make sure the pendulum shaft does not slide out during the rotational movement of the pendulum arm, external retaining rings are used to secure the pendulum shaft. To install the external retaining rings, grooves are machined onto the top of the second and third pendulum arm shafts, shown in Fig. C.1.

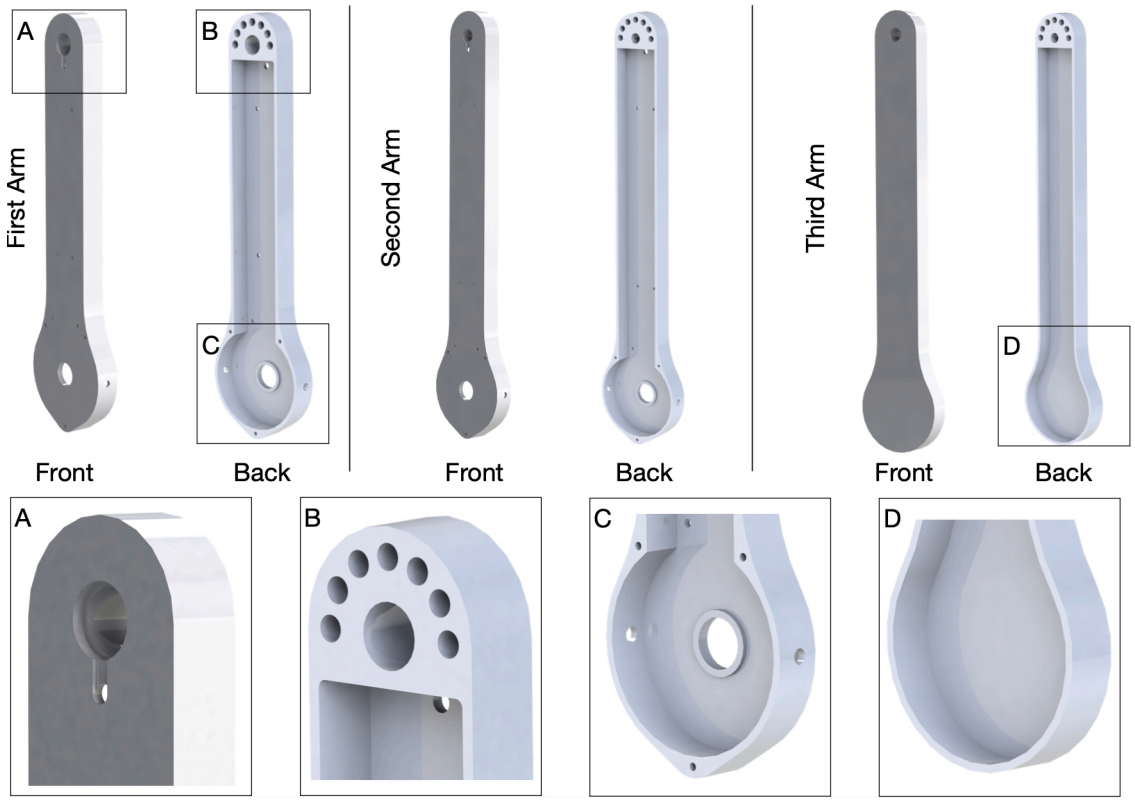


Figure C.2: Design of the pendulum arm. (A) design features necessary to allow proper wiring of the slip-ring, and the hole where the pendulum shaft is installed. (B) back side of the pendulum where the shaft is installed, and a sequence of drilled holes that reduce the weight of pendulum arm. (C) place where the bearing and encoder are installed. (D) third pendulum arm without bearing hole.

C.2.2 Manufacturing Details

Several components of the pendulum arm are manufactured: 1) pendulum arm (Fig. C.2), 2) pendulum shaft (Fig. C.1), 3) bearing plate (Fig. C.1), 4) protection case (Fig. C.2 (E) and (F)), and 5) wire clipper (Fig. C.2 (B)). In this section, we talk about the manufacturing details of those components.

The pendulum body is CNC milled from a multipurpose 6061 aluminum block with a dimension of 5/8" thick, 3" wide, 1 foot long. The aluminum block is first face milled on the front and back faces. Then the edge of the aluminum block is machined. This step

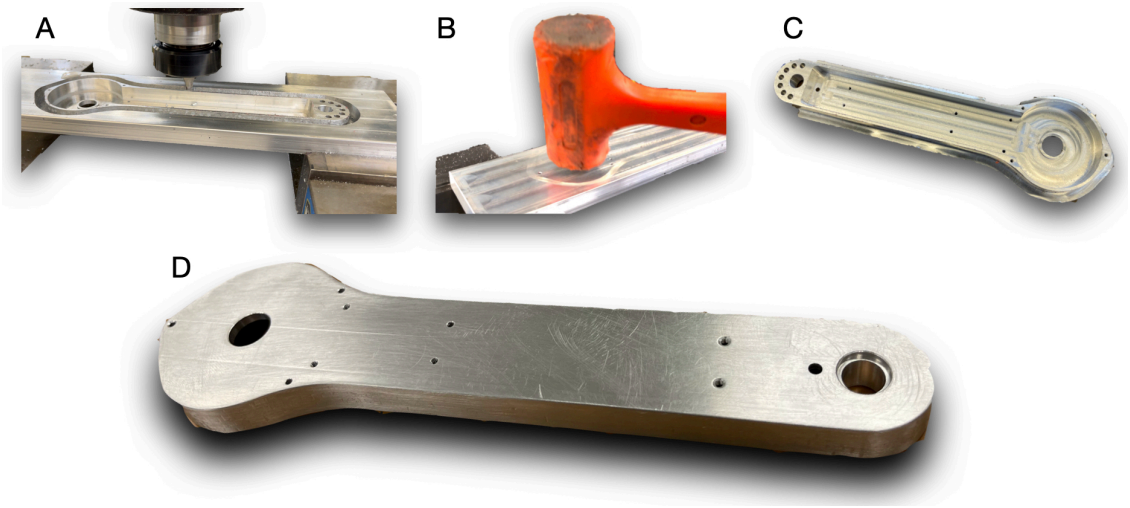


Figure C.3: Overall steps needed to manufacture the pendulum arm: (A) the inside features of the pendulum arm is machined, and the outline of the pendulum arm is processed until only a thin layer of aluminium connects the pendulum arm and aluminium block. (B) and (C) the machined pendulum arm is knocked off using a robber hammer. (D) the machined pendulum arm is polished to get rid of any burr on the edge of the pendulum arm.

allows to make the faces of the aluminum block parallel to each other (front and back, edge to edge). Next, the aluminum block is secured on the CNC machine, and the center of the shaft hole is used as the origin of the x, y, and z coordinates. The back side of the pendulum arm is first machined. While more material remains in the aluminum bar, the hole for the pendulum shaft is first drilled to a diameter close to the actual diameter needed. Next, the CNC mill is used to refine the shape and dimension of the hole until it is close to the desired dimensions. Finally, the reamer is used to refine the size of the hole for a smooth installation of the shaft. We choose to first machine the hole for pendulum shaft when there is more material left, since this will increase the rigidity of the aluminum bar during the drilling and milling process, resulting in a more precise hole. After manufacturing the hole for the shaft, the hole for the bearing installation is machined using similar steps, where the hole is first drilled, then milled, and finally refined using a reamer. Next, the threaded hole for connecting the bearing plate and the pendulum arm is drilled, as shown

in Fig. C.2 (C). After this step, the weight reducing hole shown in Fig. C.2 (B) is machined. Next, the aluminum block is flipped and recenter using edge finder. Then, the hole shown in Fig. C.2 (A) is milled to correct shape and depth. Next, a threaded hole is drilled to install the wire clipper and encoder, shown in Fig. 6.4 (B) and Fig. C.2 (C). Then the pendulum arm is flipped again to the back side to machine the inner part. The inner side of the pendulum arm is milled by programming the CNC machine to achieve the desired shape, shown in Fig. C.2 [1]. Once this step is finished, the outer shape of the pendulum arm is milled, shown in Fig. C.3 (A). The TiAlN Coated, 2 flute, 1/4" mill diameter, 2-1/2" overall length end mill is used to mill the outer shape of the pendulum arm. At the final round of machining, a thin layer (0.5 mm) of aluminium is left to connect the pendulum arm body and the remaining of aluminum block, then a rubber hammer is used to knock out the pendulum arm, as shown in Fig. C.3 (A) and (B). The resulting pendulum arm will have rough edges, shown in Fig. C.3 (C). Thus, the pendulum arm is polished after the holes on the side of the pendulum arm shown in Fig. C.2 (C) are drilled. The final pendulum arm is shown in Fig. C.3 (D).

The similar process is used to machine the bearing plate by using a 0.16" thick, 6" x 6" multipurpose 6061 aluminum sheet. After the aluminum sheet is secured on the CNC machine, the face mill is used to get the correct height of the bearing plate. Next, the bearing hole is manufactured. Then the through holes shown in Fig. C.1 are drilled. Finally, the outer shape of the bearing plate is milled until there is only a thin layer of aluminum connecting the bearing plate and aluminum sheet, and the bearing plate is knocked out using the same approach shown in Fig. C.3 [2].

The material we used for the pendulum shaft is 1566 carbon steel, which balance high strength and good machinability. For the first pendulum arm, 12" long material with 1/2" diameter raw material is used, while 12" long 3/8" diameter raw material is used for the

[1]The inner side corner of the manufactured pendulum arm will not be a straight corner. Instead, its shape is an arc whose radius is determined by the radius of the end mill used to machine the inner side.

[2]Same as footnote 3, the straight corner in Fig. C.1 will be an arc whose radius is determined by the radius of the end mill.

second and third shaft. To manufacture the shaft, a lathe is used. When machining the first and second arm shaft, a through hole is first drilled. Then, a shallow hole is drilled to create a step which allows proper installation of the slip-ring. This can be seen in Fig. C.1. Next, the main diameter of the shaft is lathed. After the major shape of the shaft is obtained, a groove is lathed for the second and third arm to allow installing the external retaining ring. For the first arm, a thread is lathed to allow application of the axial force by screwing a nut. The groove on the bottom of the first and second shaft is milled after it is press fitted into the pendulum arm body. After the machining, all the parts are polished and cleaned to remove dust or metal chips. Finally, the protection case and wire clip is 3D printed using Polylactic Acid (PLA) material. The main functionality of the case is to protect the slip-ring inside the pendulum arm from accidental collision.

C.2.3 Assembly Details of Second and Third Arm

The assembly of the double pendulum is described in section 6.3. For the assembly of the triple pendulum illustrated in Fig. 6.6, the following steps are needed: 1) Follow all the assembly steps of the double pendulum. 2) Slide the shim (11) into the shaft of the third arm (22). Next, slide the third pendulum shaft (22) into the bearing of the second arm (12) and stop until the shim (11) contacts the inner ring of the bearing. 2) Slide the shim (16), encoder disk (17), and another shim (18) onto the third arm's shaft (22). Then, slide the assembled bearing plate onto the shaft (22). Finally, clip the external retaining ring (19) onto the shaft (22). This step is the same as mentioned before. 3) Slide the 3D printed shim (6) and encoder reader (7) into the desired place by aligning the holes. Then, install the screws (8) to secure its position. 4) Next, install the protection case by aligning the holes on the pendulum arm body, bearing plate, and protection case. Mount the screws (4) to fasten the assembly. 6) Finally, install the wire clipper (5) into the pendulum arm (15) by aligning the holes and tighten the screws (4). The above steps finish the assembly of the second and third pendulum arm, and results in the final assembly shown in Fig. 6.4. The

complete assembly process of the second and third arm are illustrated in Fig. 6.6.

C.3 Pendulum Cart Details

Our multi-link pendulum uses a linear motor to provide actuation to the pendulum arm. The advantage of using the linear motor is that it does not have backslash issues, which frequently happen in the belt-drive type servo motor. This benefit allows accurate control of sensitive maneuvers, such as the swing-up of double and triple pendulums. As discussed in section 6.4, to size the linear motor, first, the desired maximum speed and acceleration of the cart are defined using desired motion profile of the pendulum. We choose a HIWIN linear motor system LMX1K-SA12-1-2000-PGS1-V103+HS. For more details on choosing the linear motor size, we refer the readers to see Appendix A of the HIWIN linear motor system manual [471]. The overall look of the selected linear motor can be seen in Fig. C.4. As Fig. C.4 shows, the linear motor already comes with a cable management track and motor stage. Moreover, Fig. C.4 (A) suggests the motor stage is not flat, which suggests the proper design of connection between the pendulum arm and linear motor stage is needed. Fig. C.4 (B) indicates there are pre-drilled holes on the backside of the linear motor stage. Those pre-drilled holes can facilitate the installation of a switch plate. Finally, Fig. C.4 also indicates that the both side of the linear motor rail has shock absorbers installed which prevents the direct collision of the motor stage and the end of linear motor rail.

The connection between the pendulum arm and the linear motor stage are shown in Fig. 6.7 (A) and (B). A bearing housing is needed to provide support for the first pendulum arm shaft and secure it so that the pendulum arm can perform free swing. Moreover, an aluminum plate is machined so that the bearing housing can be connected with the linear motor stage. The details of the bearing housing can be seen in Fig. C.5. As Fig. C.5 shows, one thorough hole is drilled on the front and back side of the housing. This hole is used to install the bearings that mate with the first pendulum shaft. In order to maintain the relative distance of the bearings, a spacer needs to be installed. Same as the pendulum

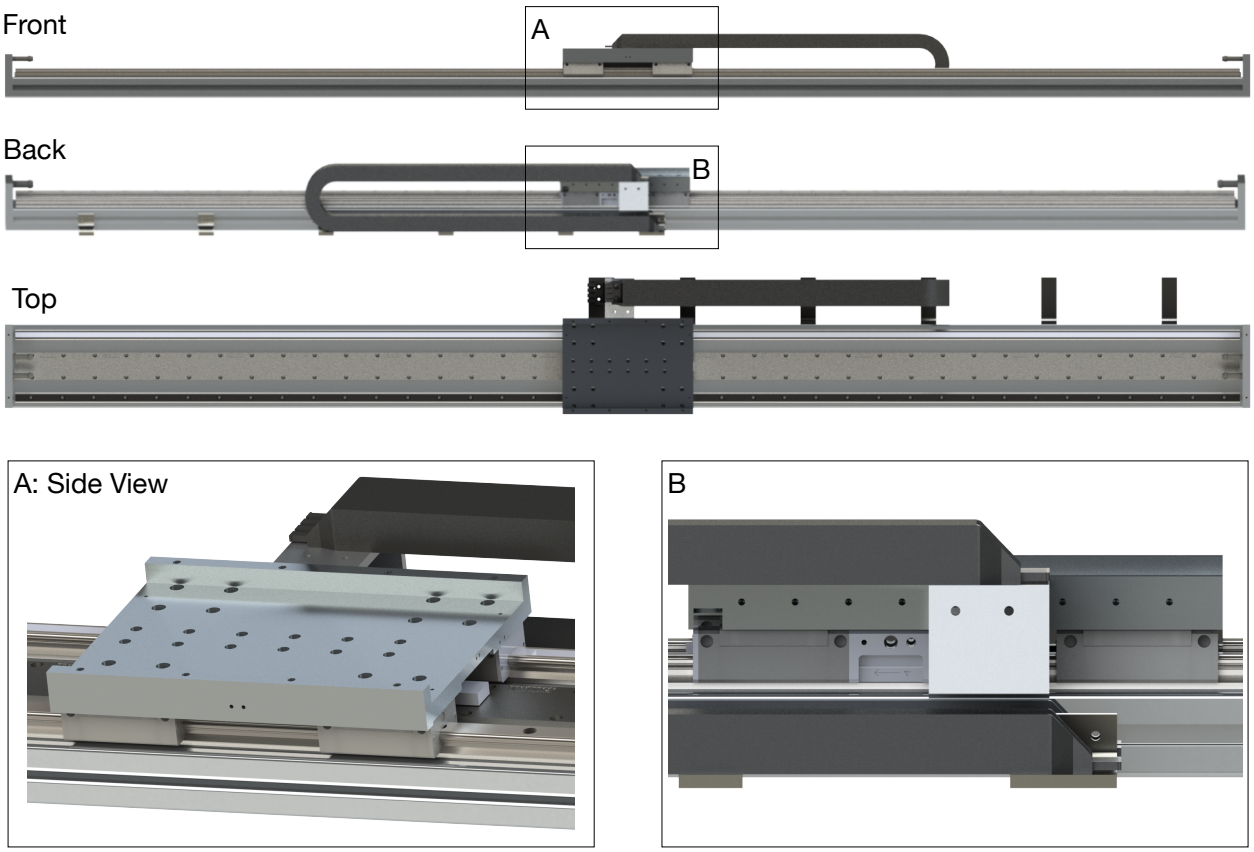


Figure C.4: Overview of the linear motor that drives the pendulum cart, with the linear motor cable track and the stage with pre-drilled holes, which allow the installation of user designed parts. This linear motor stage is used to mount the aluminum plate that is used to install the bearing housing and the pendulum arm. (B) Limit switch plate installed on the back side of the linear motor stage.

arm, the ceramic bearing is used to minimize the drag force. Four more thorough holes are been drilled on the bottom of the bearing house which is used to mount the bearing housing to the aluminum plate and linear motor stage. Two threaded holes are drilled on the side panel of the bearing house and are used to mount the encoder reader. To avoid collision of the encoder disk and bearing housing, the bottom of the bearing housing has been designed as a C shape. As for the aluminum plate, the design is straightforward as Fig. 6.7 and Fig. C.5 shows. It has four holes that is used to mate the aluminum plate with

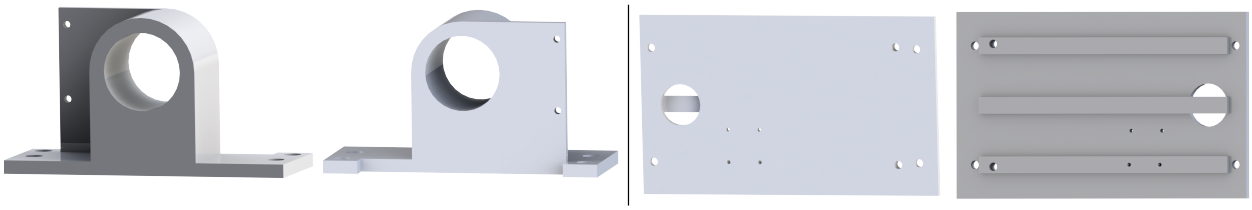


Figure C.5: This figure illustrates the overall look of the bearing housing and cart plate. The main features on the bearing housing is the thorough hole that is used to install the bearing. To make the manufacturing process easier, a thorough hole is used and a spacer is later installed to maintain the distance of two bearings. As for the cart plate, it has simple shape and features. The four tiny threaded holes are used to install the cable management tool while the larger hole is milled to install the circular level indicator.

linear motor stage. The back side of the aluminum plate has multiple stripes that acts as a stiffener which is used to strengthen the plate. Four small threaded holes are drilled on the top of the aluminum plate to install 3D printed cable managing tool while one bigger hole is milled to install the circular level indicator. This circular level indicator is useful as it can help user to level the pendulum cart. The final piece we designed for the pendulum cart are the limit switch plate as it shown in Fig. 6.7 (B). Fig. 6.7 (B) shows two limit switch plates installed on the back side of the linear motor stage, they are responsible to block the laser limit switch when the pendulum cart moves to the edge of the linear rail. The laser limit switch is installed on the linear motor by designing a 3D printed base as Fig. 6.9 (B) shows.

C.3.1 Manufacturing Details

The major parts that need to be manufactured in the pendulum cart are: 1) Bearing housing. 2) Pendulum cart plate. 3) Limit switch plate. 4) 3D printed cable management tool. 5) 3D printed limit switch base. This section will go through the manufacturing process of those components.

To manufacture the bearing housing, a 3" multi-purpose aluminum cube is machined.

Fig. C.6 shows the overall machining process.

1) To start with, the raw material is face milled on all fronts to create datum planes. Next, a shallow hole whose diameter is smaller than the diameter of the bearing is milled, and the center of the hole is set as the origin of the coordinate system. Then, the front of the bearing housing is milled to create the wanted shape. The milling continues until the end mill reaches the same height as the side panel used to mount the encoder reader. 2) Once the front shape of the bearing housing is machined, it is flipped over to manufacture the features on the backside. The aluminum cube is first milled until the end mill is at the same height as the total width of the bearing housing. Next, a through-hole is drilled whose diameter is smaller than the bearing to be installed. Then a reamer is used to refine the shape and diameter of the hole. The goal is to make the bearing hole and bearing have a transitional fit. Once this step is done, the milling is continued until the backside of the bearing housing is machined to desired shape. Then two threaded holes are drilled for installing the encoder reader. When drilling these two holes, the origin of the bearing hole is used as the center of the coordinated system since the relative distance between the bearing hole and the holes for installing the encoder reader is what matters. 3) Finally, the bearing housing is milled in the upright position to produce a straight corner. Once finished, it is polished using sandpaper, and the sharp edges are smoothed using a deburring wheel. It is noteworthy that this bearing housing can be 3D printed if it does not have too much load on it, for example, when only a single pendulum formulation is needed.

The manufacturing of the pendulum cart plate is easy to do. A multipurpose 6061 Aluminum with 7/16" thickness and 8" x 8" length and width is used to manufacture the pendulum cart plate. The following steps are needed to manufacture it. 1) To start with, the raw material is face milled to create a datum plane. Next, it is milled to have the same width and length as the designed cart plate. 2) Next, the front face of the cart palate is face milled, and all the holes needed are drilled. This process includes the hole used to install the circular level indicator. 3) Once the features on the front of the cart plate are

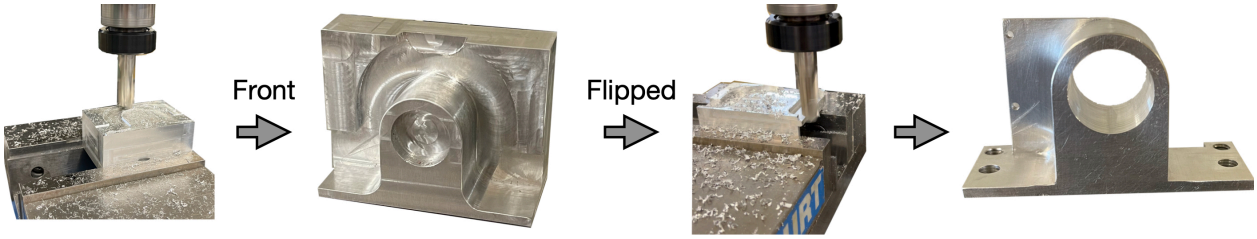


Figure C.6: This figure illustrates the overall process of machining the bearing housing. To start with, the aluminum cube is face milled to create the datum face. Then the features on the front of the bearing housing is machined. Next, the aluminum cube is flipped over to machine the features on the backside of the aluminum cube and all the necessary holes needed are drill. Finally, the bearing housing is polished and sharp edges are smoothed.

manufactured, it is flipped over to mill the stiffener. Finally, the post-processing is done to polish the cart plate, and the deburring wheel is used to make the smooth edges. If the pendulum cart does not have a considerable load and does not require high-speed movement, it is also possible to 3D print the pendulum cart plate.

The left and right limit switch plates are manufactured using a multipurpose 6061 aluminum sheet with 0.016" thickness. The overall outline of the limit switch plate is first drawn on the aluminum sheet using a marker pen and ruler. Then, the holes are drilled, which are needed to install the plates to the linear motor stage. Then, a metal brake and scissors cut the aluminum sheet to the desired shape. Finally, the aluminum sheet is bent, polished, and deburred. After using the above steps, the switch plates can be produced easily. As for cable management tools and limit switch base, they are 3D printed with PLA filament.

C.4 System Frame Details

The linear motor selected in Sec. 6.4 is mounted on a support frame in order to minimize the unwanted vibration during the cart movement. This motivates us to design a system frame to mount the pendulum cart and provide support to it. This section introduces the design and manufacturing of the system frame we used. Moreover, we show how to

install the linear motor to the system frame and provide a detailed bill of materials for reproduction.

C.4.1 Design Details

Fig. C.7 shows the overall view of the system frame we designed. Our design is similar to the one shown in [429, 448]. To make sure the pendulum arm won't collide with the ground, four vertical aluminum extrusions are connected with the linear motor using corner concealed brackets. This lifts up the linear motor from the ground and creates space for the pendulum arm. To make sure the vertical aluminum extrusion is stable in the $x - z$ plane, the diagonal brace is used to connect the linear motor and vertical aluminum extrusion as shown in Fig. C.7 (A). Moreover, four more aluminum extrusions are connected together using a machined connector plate to create a X shaped frame as shown in Fig. C.7 (C). By connecting the X shaped frame and the four vertical aluminum extrusions, the movement of the frame is further constrained in the $x - z$ plane. Next, the frame's movement is constrained in the $y - z$ plane. This is achieved by connecting the vertical frames using an aluminum extrusion as shown in Fig. C.7 (B) and (D). To provide more support, the vertical aluminum extrusion on the back side is further connected to another aluminum extrusion to form a triangle using T-slotted framing structural brackets, as shown in Fig. C.7 (E) and (F). To provide a space to place the electrical box, the real-time system and the development system, two horizontal aluminum extrusions are used to connect the left and right vertical aluminum extrusion as shown in Fig. C.7 (F). Finally, to allow easy movement of the experimental system from place to place, four wheels can be installed as an option which is shown in Fig. C.7 (F). However, it is recommended that the wheels are only installed when it is necessary to move the system frames. When performing actual experiments, the wheels should be removed to avoid unwanted oscillation of the system frame.

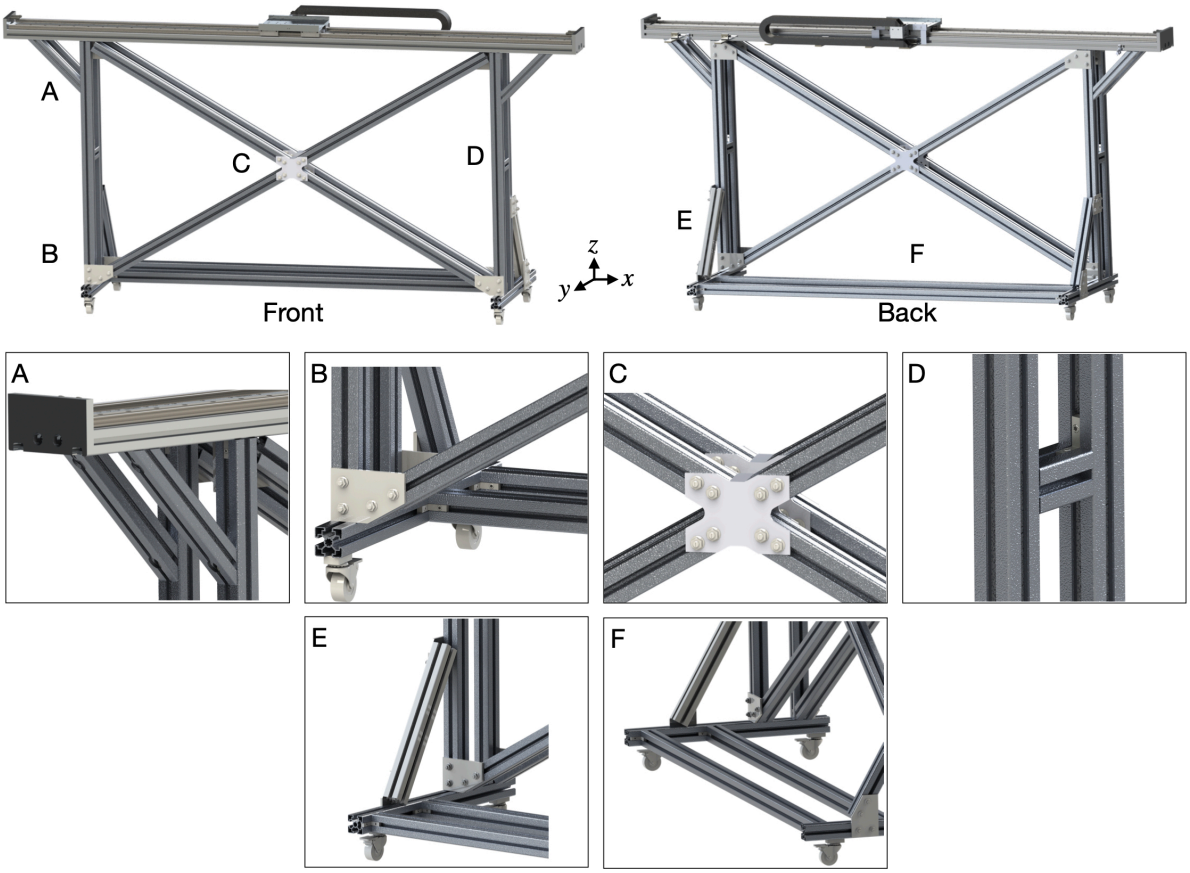


Figure C.7: System frame. The linear motor is lifted up using four vertical aluminum extrusion. Different aluminum extrusions are installed to constrain the movement of system frame in the $x-z$ and $y-z$ plane. To allow easy moving of the experimental setup, four wheels can be installed to the system frame as shown in (E) and (F).

C.4.2 Manufacturing Details

The manufacturing of the system frame is straightforward since most of the components can be directly purchased online. The main components that need to be manufactured are: 1) Aluminum extrusion, which has to be trimmed down to the correct length. 2) Aluminum extrusion connector plate, which needs to be machined. 3) 3D printed alignment tool, which is used to facilitate the assembly of the X shaped frame. In order to trim the aluminum extrusion to the correct length, the band saw is first used to trim the aluminum

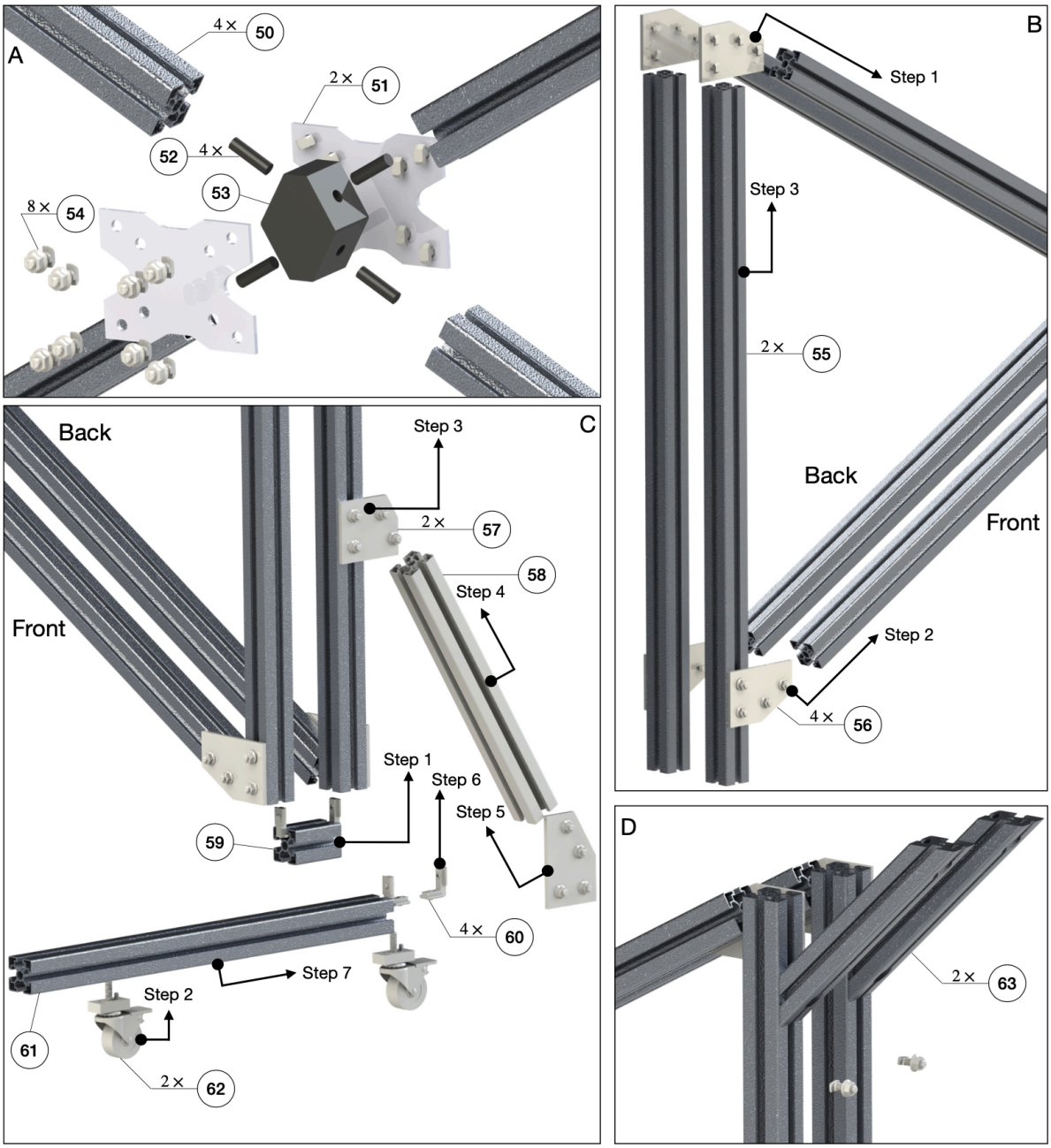


Figure C.8: Assembly of the system frame. (A) Installation procedure of the X shaped frame. (B) Connection of the X shaped frame with the vertical aluminum extrusion (55). (C) Installation of the wheel to the system frame. (D) Installation of diagonal brace.

extrusion to a length that is slightly longer than the desired length. Next, the CNC mill is used to machine the cut surface of the aluminum extrusion to the correct length. Finally,

the machined end is polished and unwanted burr is removed. To machine the connector plate shown in Fig. C.7 (C), a 0.16" thick, 6" x 6" aluminum plate (89015K255) is machined. First, the through holes that is used to mate with the bolts and nuts are drilled. Then, a simple CNC program is written to mill the outer shape of the connector. As for the alignment tool that helps the assembly of the X frame, it is printed using PLA filament.

C.4.3 Assembly Details

To assemble the system frame, four major steps are needed: 1) The X shaped frames are assembled. 2) The X shaped frame is connected with the four vertical aluminum extrusions. 3) The assembled aluminum frame is connected with the aluminum extrusion in the $x-y$ plane and diagonal braces for single rails. 4) Finally, the linear motor is installed with the system frame and all the screws are fastened accordingly. The above mentioned steps are illustrated in Fig. C.8.

As Fig. C.8 (A) shows, to better assemble the X shaped frame, a 3D printed alignment tool ($52, 53$) is used. The alignment tool is first assembled by inserting the 3D printed cylinder (52) into the 3D printed block (53). The 3D printed block (53) serves as a reference to make sure each aluminum frame (50) has the correct angle to each other. This is achieved by inserting the aluminum frame (50) into the 3D printed cylinder (54), making sure it is perpendicular to the surface of 3D printed block (53). Once the correct angle between the aluminum frame (50) is achieved, the connector plate (51) is used to connect each frames using drop-in bolts and nuts (54). This finishes the step 1, assembly of the X shaped frame.

In step 2, 60 degree bracket (56) is first slid into the aluminum frame (50) as shown in Fig. C.8 (B) step 1 and step 2. Then the vertical aluminum frame (55) is slid into the 60 degree bracket as shown in step 3 of Fig. C.8 (B). This completes the assembly of the X shaped frame and vertical frame. To avoid the collision of the pendulum arm and 60 degree bracket (56), the top right and top left 60 degree bracket (56) is installed on the back side of front X shaped frame, as shown in Fig. C.8 (B).

Fig. C.8 (C) and (D) illustrates the details of step 3. First, the corner concealed bracket (60) is slid into the aluminum spacer (59). Then, this aluminum spacer is slid into the middle of front and back X shaped frame as shown in Fig. C.8 (C) step 1. In step two, the wheel (62) of the system frame is assembled with the aluminum frame (61). In step 3, the 30 degree bracket (57) is first slid into the vertical aluminum frame of the back X shaped frame (55) and the bolts and nuts are slightly tightened to keep them in place. Then, step 4 is conducted in which the aluminum frame (58) is slid into the 30 degree plate (57). Next, in step 5 a 60 degree bracket (56) is slid into the aluminum frame (58) and slightly tightened. In step 6, the corner concealed bracket (60) is installed with the vertical aluminum frame (55) and in step 7 the aluminum frame (61) is slid into the corner concealed bracket (60) and 60 degree plate (56). Next, the diagonal brace in installed into the vertical frame as shown in Fig. C.8 (D), and the screws are slightly tightened to make sure the diagonal brace stays in place. The procedure to install the horizontal aluminum extrusion (64) to the aluminum extrusion with wheel (61) can be seen in Fig. C.9 (A). As Fig. C.9 (A) shows, the corner concealed bracket (60) is first installed into the aluminum extrusion (64) then extrusion (64) is slid into the extrusion (61). This further stabilizes the system frame and extrusion (64) can be used to place the development computer and electrical box, as shown in Fig. 6.2. The above steps finish the assembly of the system frame. After the system frame is assembled, the linear motor (30) can be installed onto the system frame shown in Fig. C.9 (B). To do so, the corner concealed bracket (60) is installed onto the aluminum base of the linear motor. Then the corner concealed bracket (60) is slid into the vertical aluminum frame (55). Next, the diagonal brace is connected with the linear motor. Finally, all the screws and nuts are tightened. Since the linear motor and aluminum frame can be heavy, all the installation process mentioned above should be finished by at least two persons.

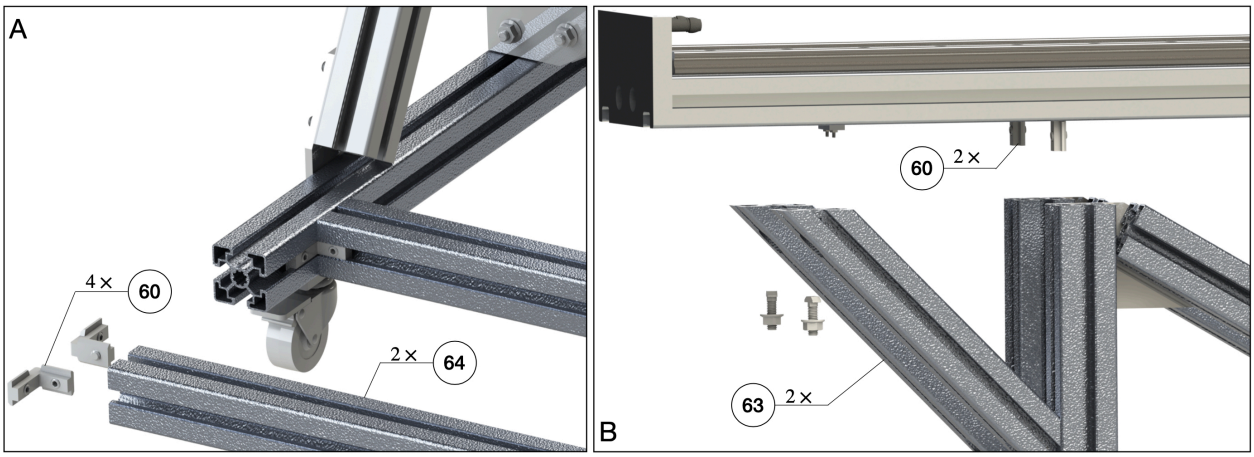


Figure C.9: Assembly process of the system frame (A) and linear motor (B).

C.5 Electrical System Details

C.5.1 Pendulum Arm Wiring Details

Fig.C.10 and Table. C.1 show the detailed wiring specification of the pendulum arm. Fig.C.10 shows the first encoder is directly connected to the first differential driver while the second and third encoders are connected with the differential driver using slip-rings. Moreover, the output of the differential driver connects to the US Digital CA-C10-SH-NC 10 feet cables, which then connect to the target computer terminal board as Table. C.1 suggests. It is important to point out that the US Digital CA-C10-SH-NC 10 feet cables are shielded cables. Thus, the shield also connects to the ground of the IO-392 ground pin, which helps reduce the effect of EMI noise.

C.5.2 Components Choice of Linear Motor's Electrical System and Wiring Details

This section will detail the functionalities and specifications of the electrical parts used to power the linear motor. A complete wiring diagram of the entire system is also provided.

The first component introduced here is the emergency stop button module. During the multi-link pendulum experiments, the linear motor may perform undesired motion due

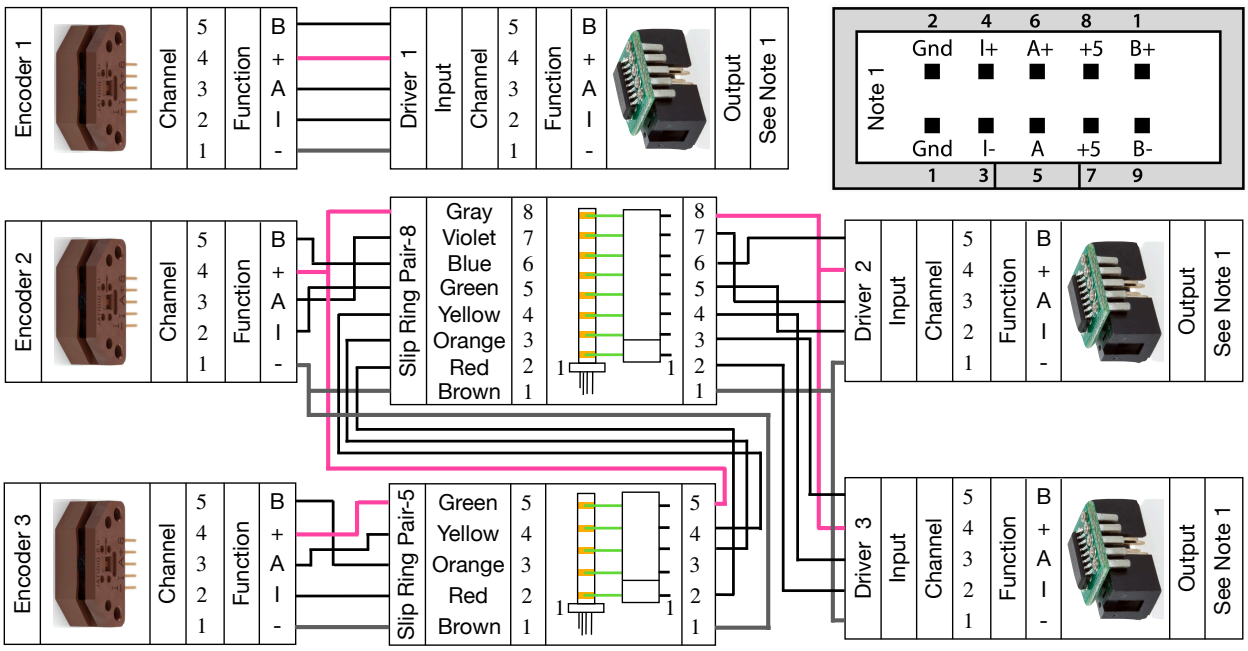


Figure C.10: This figure illustrates the overall electrical wiring of the pendulum arm encoder. The first encoder is directly connected to the first differential driver. To connected the second and third encoder to the differential driver, two slip-rings (slip-ring and brush block) are used. The second and third differential driver shares the 5V and ground channel of the first slip-ring. Moreover, the first slip-ring's 5V and ground channel is connected to the second slip-ring, which then conducts electricity to the third encoder. The third encoder uses the both the first and second slip-ring to connect with the differential driver. The output of the differential driver is connected to the US Digital CA-C10-SH-NC 10 feet cables, and those cables are then connected to the IO-392 module as Table. C.1 shows.

to coding error or controller failure. It is also possible that unwanted electrical shortage happens due to the improper use of the system. In those undesired situations, the user must press the emergency stop button to cut all the power supplied to the linear motor, avoiding damage to the setup and protecting the operator around the equipment. It is also necessary to have an On/Off button that controls the system's power supply. The POWERTEC 71354 magnetic switch (75) is selected as our emergency stop button to meet those requirements. This emergency stop button can be used in a single-phase 120V power line, matching how we supply the linear motor power. By checking the specification of the

Table C.1: This table shows the pin out of IO-392 module for encoder reading and its connections with the differential driver. D1 to D3 represents the differential driver 1 to 3 shown in Fig. C.10. "-x" represents the corresponding "x"-th pin of the specific electrical component. The ground of the encoder sensor is connected to "0 V" instead of "ground" as suggested by Speedgoat.

Pin	Code Module Channel	Functionality	Direction	Transceiver	Connects To
1a		Ground			
2a		Ground			
3a	1	QAD-A	IN	RS422/485(+)	D1-6
4a				RS422/485(-)	D1-5
5a	1	QAD-B	IN	RS422/485(+)	D1-10
6a				RS422/485(-)	D1-9
7a	1	QAD-C/Index	IN	RS422/485(+)	D1-4
8a				RS422/485(-)	D1-3
9a	2	QAD-A	IN	RS422/485(+)	D2-6
10a				RS422/485(-)	D2-5
11a	2	QAD-B	IN	RS422/485(+)	D2-10
12a				RS422/485(-)	D2-9
13a	2	QAD-C/Index	IN	RS422/485(+)	D2-4
14a				RS422/485(-)	D2-3
15a		0 V			D1-2, D2-2, D3-2, Differential Cable Shield
16a		5 V			D1-7, D2-7, D3-7
17a		Ground			
1b		0 V			
2b		5 V			
3b	3	QAD-A	IN	RS422/485(+)	D3-6
4b				RS422/485(-)	D3-5
5b	3	QAD-B	IN	RS422/485(+)	D3-10
6b				RS422/485(-)	D3-9
7b	3	QAD-C/Index	IN	RS422/485(+)	D3-4
8b				RS422/485(-)	D3-3
9b	4	QAD-A	IN	RS422/485(+)	HIWIN D1-CN2-16
10b				RS422/485(-)	HIWIN D1-CN2-17
11b	4	QAD-B	IN	RS422/485(+)	HIWIN D1-CN2-18
12b				RS422/485(-)	HIWIN D1-CN2-19
13b	4	QAD-C/Index	IN	RS422/485(+)	HIWIN D1-CN2-20
14b				RS422/485(-)	HIWIN D1-CN2-21
15b	1	Interrupt Input	IN	RS422/485(+)	
16b				RS422/485(-)	
17b		Ground			

linear motor, we find out the peak current of the linear motor is around $12.7A_{rms}$, which is lower than the rated current of the On/Off switch ($16A_{rms}$) and emergency switch ($18A_{rms}$) installed inside the emergency stop button module. Thus, this emergency stop module is selected to control the power supply of the linear motor. The emergency stop button module should be installed somewhere that is easy to reach for the operator. This location may change based on the operator's preferred standing position. In our application, we decided to install the emergency button at the side of the system frame. Please check its user manual for details on the position holes of the module.

The next vital component is the circuit breaker (MCCB, 76). The circuit breaker has the following functionalities: 1) Circuit breaker can cut off the power supply to the linear motor when there's a shortage in the electrical system. 2) It can stop the power supply to the linear motor when it is overloaded, which avoids damage to the linear motor. 3) It can simply serve as an On/Off switch that controls the power supply to the linear motor. The first and third functions of the circuit breaker are easy to understand, and we will illustrate the second functionality more. The overloaded situation might happen when the control algorithm requires the linear motor to perform unrealistic acceleration or de-acceleration. For example, during the feedforward swing-up control of the double or triple pendulum, a feedback force is usually needed to compensate for noise and model uncertainty. It has been shown [7, 428] that sometimes the LQR gain calculated to stabilize the feedforward trajectory is near singular. This indicates a considerable control input needs to be applied to the system to compensate for a slight deviation. Suppose the feedback control force is not thresholded. In that case, the controller may require the linear motor to apply unrealistic force to the pendulum arm, causing a massive amount of currents supplied to the linear motor resulting in overload. Another case where overload might happen, although unlikely in our case, is that the linear motor gets stuck mechanically. In this situation, the motor drive will try to apply more current to the linear motor to make it move, eventually causing over current. Thus, having a circuit breaker is crucial in controlling the linear motor for safety reasons. The Eaton miniature circuit breaker FAZ-B3-2-NA (76) is

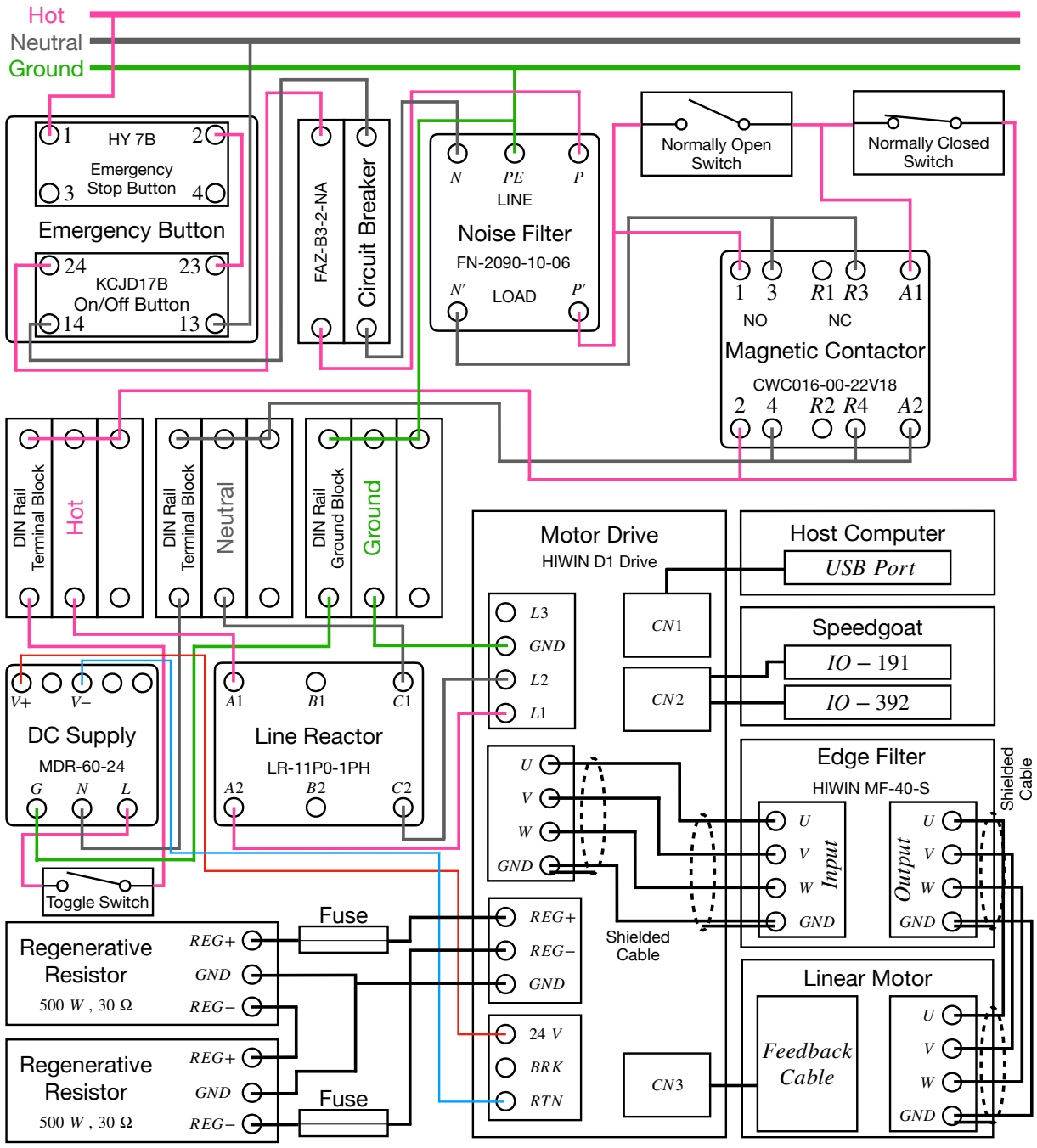


Figure C.11: This figure illustrates the overall electrical components needed to provide power supply to the linear motor. An electrical box is used to put all the components together.

used in our application. According to the user's manual, its rated current is $3A$, with the maximum allowed AC voltage being $277V$. The tripping current is 3 to 5 times the rated current, with stripping time around 0.005 to 2 seconds. The resulting tripping current ranges from 9 to $15A$ and covers the peak current of the linear motor, which is $12.7A$. Thus, using this circuit breaker can help us avoid linear motor damage when the current supplied to the linear motor is larger than the allowed peak current. Moreover, the selected circuit breaker uses a trip-free design and can not be defeated by manually holding the handles at the "On" position. Another plus is that the user can easily install it on the DIN rail. For more specifications on the circuit breaker, please check its user manual.

The magnetic contactor (77) is needed to start and turn off the linear motor. The magnetic contactor is a middle path between the power source and the linear motor. It helps avoid the direct connection of the power source to the linear motor, which helps prevent the sudden change of state of the linear motor. Combined with a circuit breaker, they can be thought of as a motor starter. The WEG Electric CWC016-00-22V18 (77) is used as a magnetic contactor in our design, allowing easy installation on the DIN rail. This contactor has two normally open and two normally closed contacts rated $16A$ current input under $60Hz$ $120VAC$. This specification fits our needs since the power output in the US is $120VAC$ at $60Hz$, and the maximum current needed by the linear motor is below the rated current of the contactor. By checking the user manual of this contactor, a "locking" mechanism of the magnetic contactor is achieved by using a momentary switch. The LCLICTOP AC 660V Red and Green Momentary Switch (86) is used to accomplish this. It can operate up to 660 VAC, and the current rating is 10 A. Fig. C.11 shows the overall wiring of the magnetic contactor and momentary switch.

It is recommended to install a line reactor on the motor drive's input side to protect it from voltage spikes, surges, and transients. The LR-11P0-1PH (79) is used to achieve this. This line reactor has a rated current of $16A$ at $120VAC$. Thus, it satisfies our usage scenario in a single-phase case.

The linear motor drive (80) controls the movement of the linear motor by providing

a high-frequency signal with sharp pulse edges. Those sharp edges of the signal can introduce unwanted ground current to the entire electrical system. The ground current can even cause interference between the different electrical components. This kind of interference is not wanted, especially when an accurate reading of the pendulum arm rotational angle is needed. The interference between the linear motor and encoder can cause additional unwanted pulse width signals to appear in the encoder's A, B, and Index channels, which might result in the wrong sensor reading. This is why the differential driver is used to transfer the single-ended encoder signal into the differential encoder signal to reduce the effect of electrical noise. Except for electrical interference, EMI can also cause electrical overstress (EOS) to sensitive components [472].

A multi-stage AC/DC EMI filter SCHAFFNER FN-2090-10-06 (78) is used according to the D1 drive's user manual to reduce the ground current in the electrical system and protect all other appliances that share the same ground, such as the host and target computers and monitors. The HIWIN MF-40-S edge filter (81) is used to eliminate the noise effect further. The edge filter helps reduce the EMI noise on the output end of the motor drive. Two D1-EMC1 EMI cores (87) can also suppress the EMI noise. One EMI core is installed on the linear motor's power input cable near the motor drive end, while the other is installed on the motor cable near the linear motor's stage end. Besides using the approaches mentioned above to reduce the EMI interference, another easy way to reduce EMI interference is proper cable management. For example, make the power cable of the linear motor stay far away from the sensor cable.

A regenerative resistor is usually needed to avoid the regenerative energy of the linear motor becoming too large and causing overvoltage to the motor drive. The motion profile of the linear motor and the weight of moving mass need to be known to pick a regenerative resistor. Then the designer can use the calculation shown in the HIWIN D1 drive's user manual (see Section 8.2 of the user manual [450]) to guide the selection of the regenerative resistor. The moving mass of the linear motor is around $5kg$, and the weight of the pendulum arm and cart plate is around $0.5kg$ in our design, resulting in a

Table C.2: This table shows the pin out of IO-191 module and its detailed connection with HIWIN D1 drive's control signal channel (CN2).

Pin	Functionality	Direction	Connects To	Pin	Functionality	Direction	Connects To
1a	Analog Input 01	In		1b	Digital I/O 01	Out	HIWIN D1-CN2-3
2a	Analog Input 02	In		2b	Digital I/O 02	Out	HIWIN D1-CN2-4
3a	Analog Input 03	In		3b	Digital I/O 03	Out	HIWIN D1-CN2-5
4a	Analog Input 04	In		4b	Digital I/O 04	Out	HIWIN D1-CN2-6
5a	Analog Input 05	In		5b	Digital I/O 05	Out	HIWIN D1-CN2-7
6a	Analog Input 06	In		6b	Digital I/O 06	Out	HIWIN D1-CN2-12
7a	Analog Input 07	In		7b	Digital I/O 07	Out	
8a	Analog Input 08	In		8b	Digital I/O 08	Out	
9a	Analog Output 01	Out	HIWIN D1-CN2-24	9b	Digital I/O 09	In	HIWIN D1-CN2-13
10a	Analog Output 02	Out		10b	Digital I/O 10	In	HIWIN D1-CN2-14
11a	Analog Output 03	Out		11b	Digital I/O 11	In	HIWIN D1-CN2-15
12a	Analog Output 04	Out		12b	Digital I/O 12	In	
13a	Analog Ground	In	HIWIN D1-CN2-25	13b	Digital I/O 13	In	NC
14a	Analog Ground	In		14b	Digital I/O 14	In	
15a	0 V	NONE		15b	Digital I/O 15	In	
16a	5 V	Out		16b	Digital I/O 16	In	
17a	Analog Ground	In		17b	Digital Ground	In	HIWIN D1-CN2-1 HIWIN D1-CN2-2

Table C.3: This table shows the pinout of the limit switch cable and its connection. The HIWIN D1 drive's CN2 channel is used to provide DC current to the limit switch. The limit switch's signal is then sent to the motor drive for the out of range protection.

D-Sub 9-Pin (Female)	Color	Function	Connects To
1	Brown	V+	HIWIN D1-CN2-22
2	Green	L-1	NC
3	Yellow	Out-1	HIWIN D1-CN2-11
4	Violet	L2	NC
5	Gray	Out-2	HIWIN D1-CN2-10
9	Blue	V-	HIWIN D1-CN2-23

total moving mass of $M_t = 5.5kg$. Then, assume the linear motor is moving at $V_1 = 4m/s$ and needs to deaccelerate to $V_2 = 0m/s$ with $a = 40m/s^2$. Using the above information and following the calculation shown in the user manual, we find there's no need to use a regenerative resistor since the energy return to the motor drive is smaller than the D1 drive amplifier's absorption capacity. However, we still decided to pick two 500W 30Ω regenerative resistors (82) and connect them in serial in case the returned energy is larger

than the energy storage capacity.

The proper operation of the linear drive requires the supply of $24V$ DC voltage. Thus, the MDR-60-24 AC to DC industrial DIN rail power supply (83) is used to output $24V$ DC at $2.5A$. A toggle switch (85) is used to control the On/Off state of the DC power supply to the linear motor drive. The Hammond CSKO242410 enclosure (84) is used to install electrical components. A Hammond CSFC2424 flush cover (88) is cut down to be used as an electrical panel that mounts some of the components mentioned above. Two DIN rails (89) are installed inside the electrical enclosure to facilitate the installation of different electrical components. This completes the selection process of all the electrical components related to the linear motor. Fig. C.11 shows their wiring details.

As Fig. C.11 shows, the emergency stop button is first connected to the single-phase power supply. Two 14 AWG power cord (90) is used as the emergency stop button's power input/output cable. Next, the output of the emergency stop button is connected with the circuit breaker, which serves as over-current protection. Then, the noise filter is connected to the circuit breaker. The connection wire used for the hot and neutral line is 12 AWG wire (91), while the connection wire for the ground is 10 AWG wire (92). The output of the noise filter connects to the magnetic contactor. The magnetic contactor is wired to self-lock when the normally open momentary switch is pressed. The output of the magnetic contactor is wired to the DIN rail terminal block (93, 94) to facilitate cable management better. Then, the DC supply and line reactor input is connected to the DIN rail terminal block. The output of the DC supply is $24V$ DC, and it is used to provide power to the motor drive. A toggle switch is used to make the supply of DC voltage controllable. Then, the output of the DC supply is connected to the Wago 721-103/026-000 connector (95). The output of the line reactor is connected to the AC power port of the motor drive using a single-phase connection formulation. The connector used is Wago 721-204/026-000 (96). The U, V, W, and GND channel is connected to the corresponding channel on the edge filter using a shielded cable to reduce the EMI noise. The connector used on the motor drive is Wago 721-104/026-000 (97). The connector used in the edge filter's input port is

Phoenix Contact 1718517 (98). The output of the edge filter then connects to the linear motor's power cable, which is a shielded cable as well. The output connector of the edge filter is Phoenix Contact 1718504 (99). Two EMI cores (87) are installed to reduce the EMI noise further. One is near the motor connection end on the motor drive and the other near the motor stage end of the linear motor. Finally, two regenerative resistors are connected to the regenerative resistor port of the motor drive in serial. The connector used for the regenerative resistor port of the motor drive is Wago 723-603 (100). The reader can choose to install two fuses as well for over-current protection. This completes the power supply to the linear motor. A D1-DNT07A USB232 to RJ11 adapter cable (101) is used to set up the motor drive parameters. The linear motor's feedback cable is connected to the CN3 of the motor drive to let the motor drive read the encoder and hall sensor signal from the linear motor. The encoder signal of the linear motor, along with other I/O signals of the motor drive, is connected to the Speedgoat's terminal block. This is achieved by making a custom cable using hook-up wires. This custom cable consists of 24 AWG hook up wires (102), cable shielding sleeve (103), cable sleeve (104), 3M 10126-3000 connector (105) and its cover 3M 10326-52F0-008 (106). Table. C.1 and Table. C.2 show the connections between the CN2 of the motor drive and the Speedgoat real-time system. Section 3 of the motor drive's user manual contains the connection details of the motor drive and its pinout of control signals (CN2). The interested reader should reference the user manual if future system modifications are needed. The final piece to mention is the limit switches we used in the linear motor for out-of-range protection. A HIWIN 3-meter limit switch extension cable is used (107) to connect the limit switch and motor drive. Table. C.3 shows the pinout of the limit switch cable and its corresponding connection to the motor drive. Zip tie and wire clips are used to make proper cable management by securing the position of all the cables/wires we used.

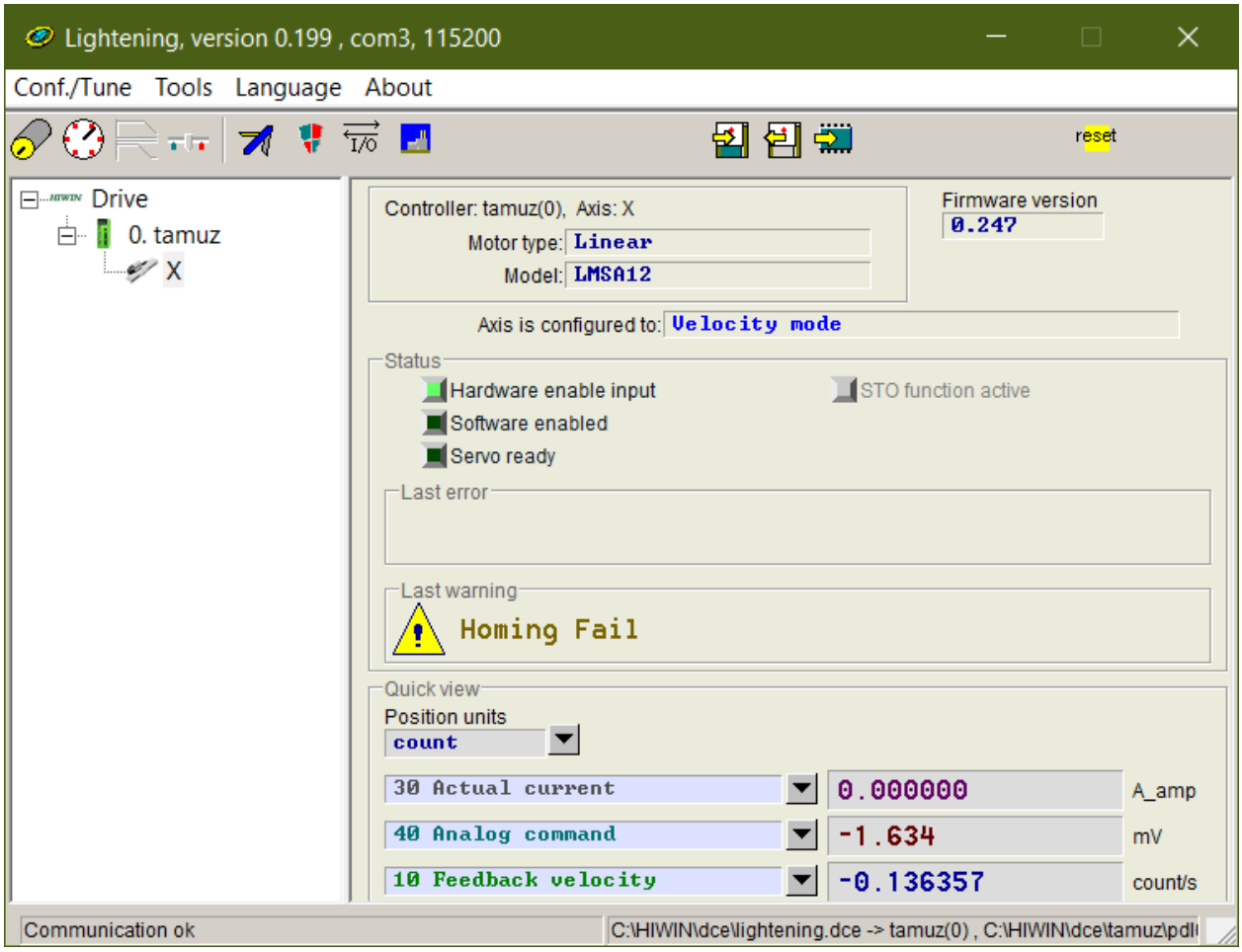


Figure C.12: This figure illustrates the overall look of the Lighting software.

C.6 Software Setup Details

C.6.1 Software Setup of Motor Drive Details

The HIWIN D1 drive's corresponding setup software is Lightning, and the reader can find a detailed introduction in sections 4 and 5 of the HIWIN D1 drive's user manual [450]. The version of the software we are using is V-0.1999, while the latest version is V-0.20 at the time of writing this paper. First, we need to download the Lightning software to the host computer to set it up. Next, the CN1 port of the D1 drive is connected to the USB port

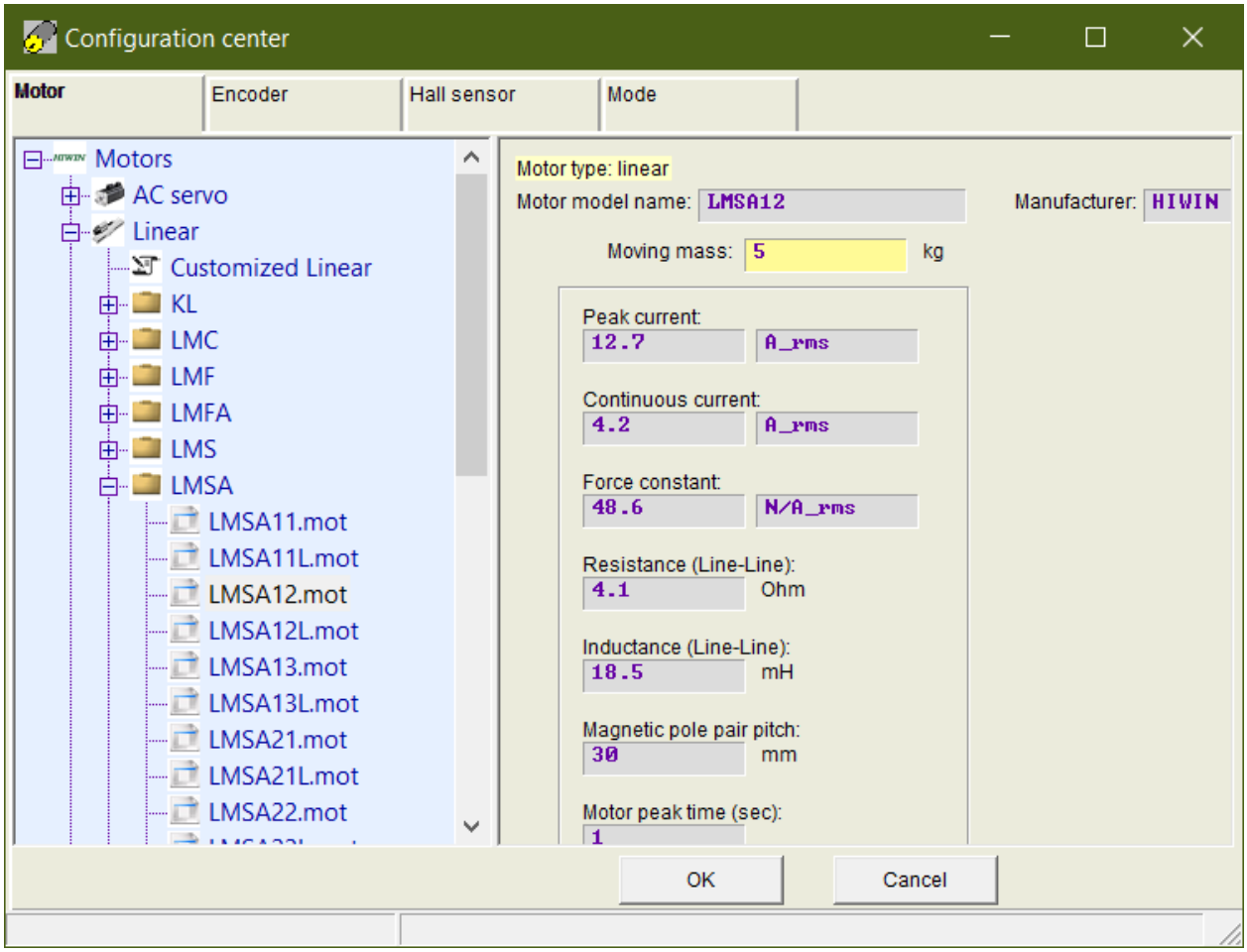


Figure C.13: This figure illustrates the configuration of the motor parameters.

of the host computer using a communication cable (101). Once finished, the D1 drive is powered up, and the Lightning software is executed. Then, in the software's menu, select "Tools->Communication setup," then configure the "BPS" drop-down list to 115200 and select the USB port that is used to connect the communication cable in the drop-down list. Finally, hit the "Apply" button, and the host computer and motor drive connection are complete. The above steps are needed to set up the necessary parameters of the motor drive. Fig. C.12 shows the overall look of the Lighting software after connecting to the host computer.

To select the correct linear motor type, motor sensor, and operating mode, we need to select "Conf./Tune->Configuration Center," which will open a new window that allows the configuration of the motor parameters and sensors. We need to first click on the "Motor" tab to select the corresponding linear motor we are using. The LMSA12 linear motor is selected in the menu box on the left. This will bring out the pre-stored motor parameters. If the linear motor from a different brand is used, the user must enter all those parameters manually. In the "Moving mass" blank, one can enter the mass of the linear motor stage. In our case, it is $5kg$. Fig. C.13 illustrates this process. After selecting the correct linear motor, we need to click on the encoder tab to set up the encoder reading options, as Fig. C.14 shows.

In Fig. C.14, the "Digital $1\mu m$.enc" is selected. The "Encoder output" panel in Fig. C.14 allows us to set up the encoder resolution output of the motor drive. Users can scale the encoder output of the motor drive to allow the successful reading of the encoder signal if another custom real-time system is used that can not read an encoder with fine resolution. This completes the setup of the linear motor encoder. Fig. C.15 shows the configuration of the hall sensor. This is achieved by clicking the "Hall Sensor" panel, and select the type of the Hall sensor as "Digital Hall sensor," and clicking on the "Enable hall phase check" option. The final parameter that needs to be configured in "Configuration Center" is the operating mode of the linear motor.

There are multiple operating modes of the linear motor. The first one is the position mode. The position mode takes the user desired position as a reference and controls the linear motor to stay at the reference position. The second mode is velocity mode. In velocity mode, the velocity of the linear motor is controlled. The user usually provides an analog signal to the motor drive, and scaling is used to transfer this analog signal to the desired velocity of the linear motor. Next, the motor drive will control the linear motor to achieve the desired speed. The third mode is force/torque mode. In the force/torque mode, the user again provides an analog voltage signal to the motor drive. Then, this voltage signal is scaled to the corresponding force/torque generated by the motor. The

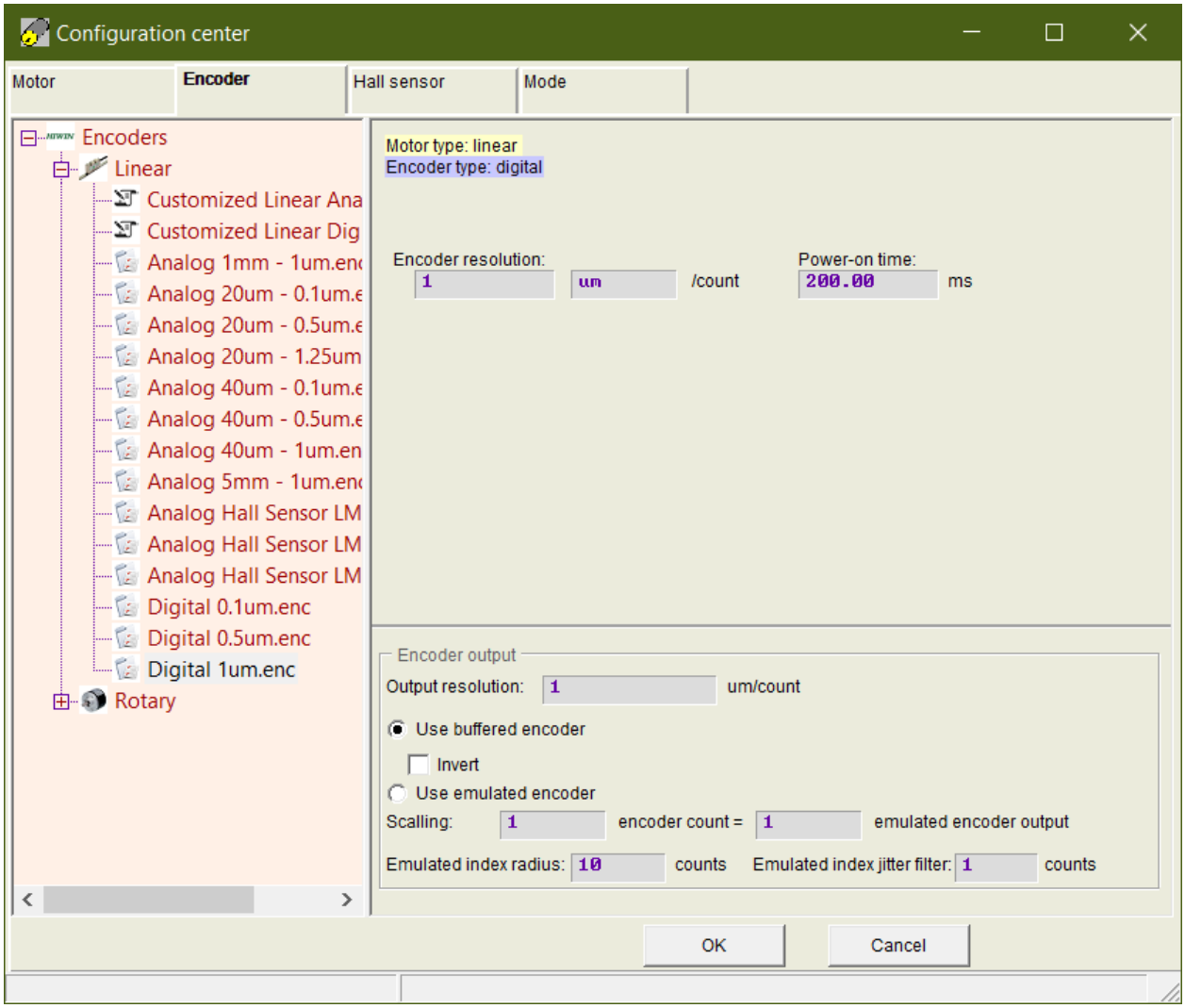


Figure C.14: This figure illustrates the configuration of the linear motor's encoder.

force/torque mode directly controls the acceleration of the motor. The final mode available is the stand-alone mode which means the linear motor can only be controlled using the Lightning software instead of the user-provided signal. In this paper, we follow the work of [7, 428] and use the velocity mode as our control method of the pendulum cart. As Appendix. C.9 shows, this results in a nicer equation of motion for the pendulum on the cart system.

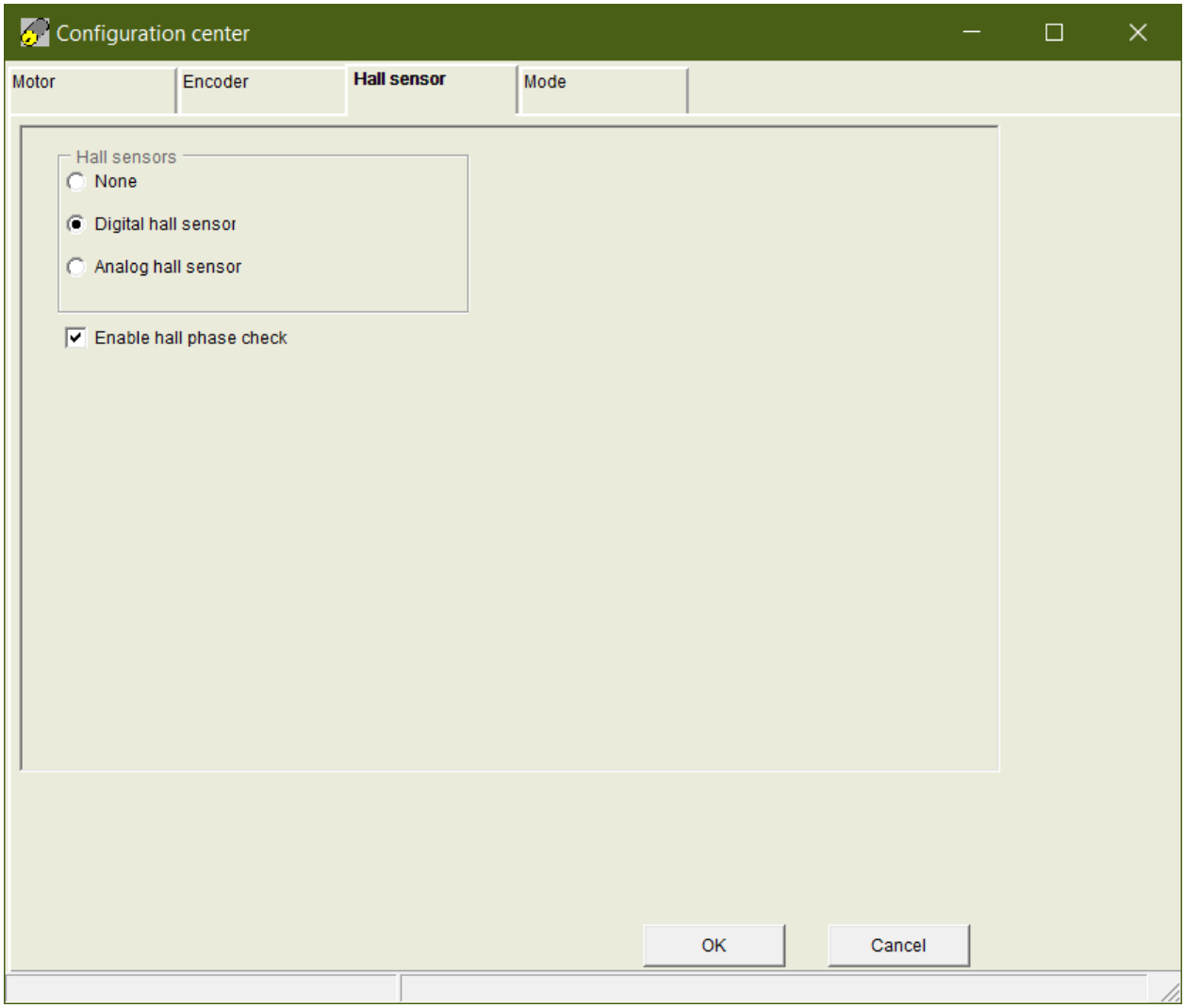


Figure C.15: This figure illustrates the configuration of the linear motor's hall sensor.

The HIWIN D1 drive supports two operating modes to be selected. The first one is set to stand-alone mode as our default operating mode. This helps avoid the unwanted movement of the linear motor when the analog signal is accidentally drifting. The secondary mode is chosen to be velocity mode, which allows us to control the velocity of the linear motor given analog input. Fig. C.16 shows the configuration of the linear motor's operating mode. During the pendulum experiments, we usually desired a speed range

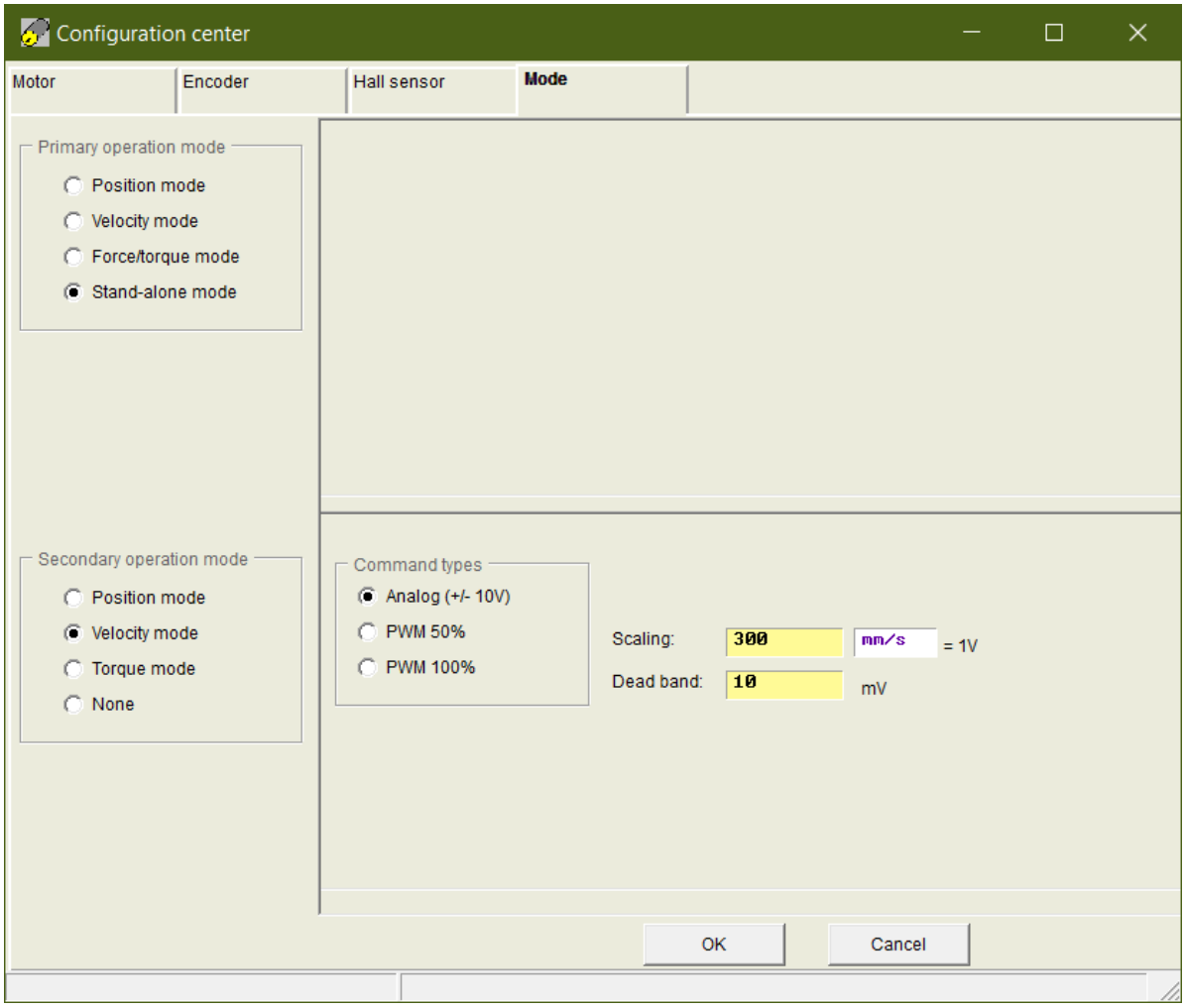


Figure C.16: This figure illustrates the selection of the linear motor's operating mode.

of $\pm 3m/s$ for the linear motor. Given the output range of the analog signal is $\pm 10V$, the scaling we used is $0.3m/s = 1V$. Moreover, we set a dead band of $10mV$. This is necessary since the analog signal is always going to be noisy. Thus, it is desired to put a dead band and filter out low magnitude noise to prevent the unwanted motion of the linear motor. The dead band we used filters out any velocity command lower than $3mm/s$.

The dead band should not be too large, or it will affect the performance of the pendulum stabilization task, especially during the stabilization of the multi-link pendulum, where

the fine movement of the pendulum cart is needed. The maximum value of the dead band that the user can select is determined by the parameters of the pendulum arm, sampling rate, scaling factor, etc. No analytical analysis is given here for adequately selecting the dead band of the analog signal. Still, a general rule to keep in mind is that the dead band should be as small as possible while eliminating most of the noise in the analog signal. The above process summarizes the configuration of the linear motor using Lightning software.

A digital input signal is needed to switch between the primary and secondary operation mode, with its high and low states corresponding to different operating modes. To set up the I/O functionality of the motor drive, the user needs to click the "Conf./Tune->I/O Center" to open the I/O setting window. Fig. C.17 shows the overall look of the configured I/O ports. In the "Inputs" tab, the I1 port of the CN2 channel (HIWIN D1-CN2-3) is configured as the hardware enable port. When the input of this channel is high, the linear motor will be hardware-enabled, and the user can achieve control of the linear motor. The I2 port of the drive (HIWIN D1-CN2-4) is configured to decide the operating mode of the linear motor. When a high signal is detected, the drive will enter the secondary mode, which is velocity mode in our case. The I3 port (HIWIN D1-CN2-5) is configured to control the homing function of the linear motor. The homing process will first drive the linear motor to the left and right limit switches. Then it will use the location information of the limit switch to find the center location between them and drive the linear motor to it. The homing function can only be activated when the motor drive is in stand-alone mode with high input received in the I3 port. Next, the I9 and I10 port (HIWIN D1-CN2-10 and 11) are configured as the limit switch signal input channel. Table. C.3 shows the detailed connection between the limit switch and CN2 channel. The above configuration helps the motor drive determine whether the linear motor reaches the limit switch. If so, the linear motor's movement beyond the limit switch is restricted. However, it will not deactivate the linear motor and stop its power supply. When an opposite-moving signal is supplied to the linear motor, it can still move in the opposite direction. This behavior is not desired, though. Once the linear motor hits the limit switches, a program should cut off all linear

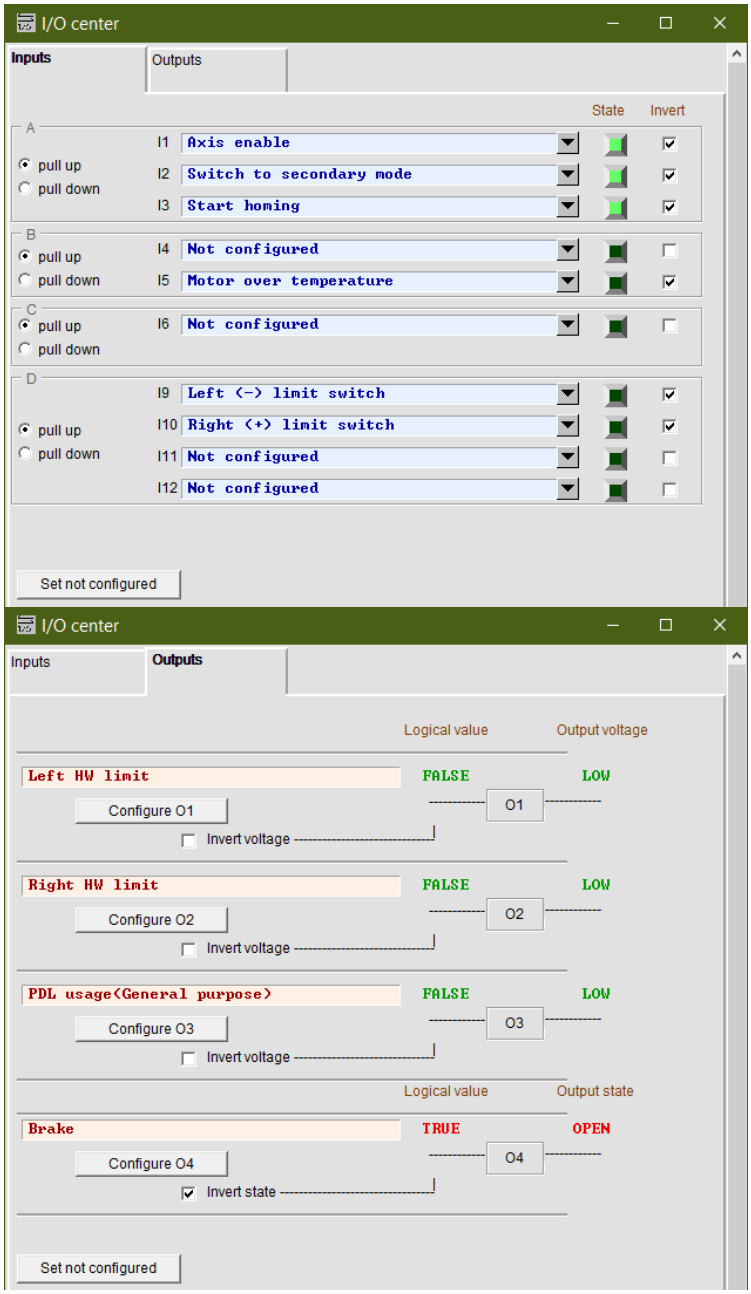


Figure C.17: This figure illustrates the configuration of the HIWIN D1 drive's I/O functionalities.

motor's power supply. The limit switch signal is supplied to the Speedgoat system to achieve this functionality. The output ports of the CN2 channel, HIWIN D1-CN2-13 and 14, are configured to be the limit switch output signal. Then, as Table. C.2 shows, the output

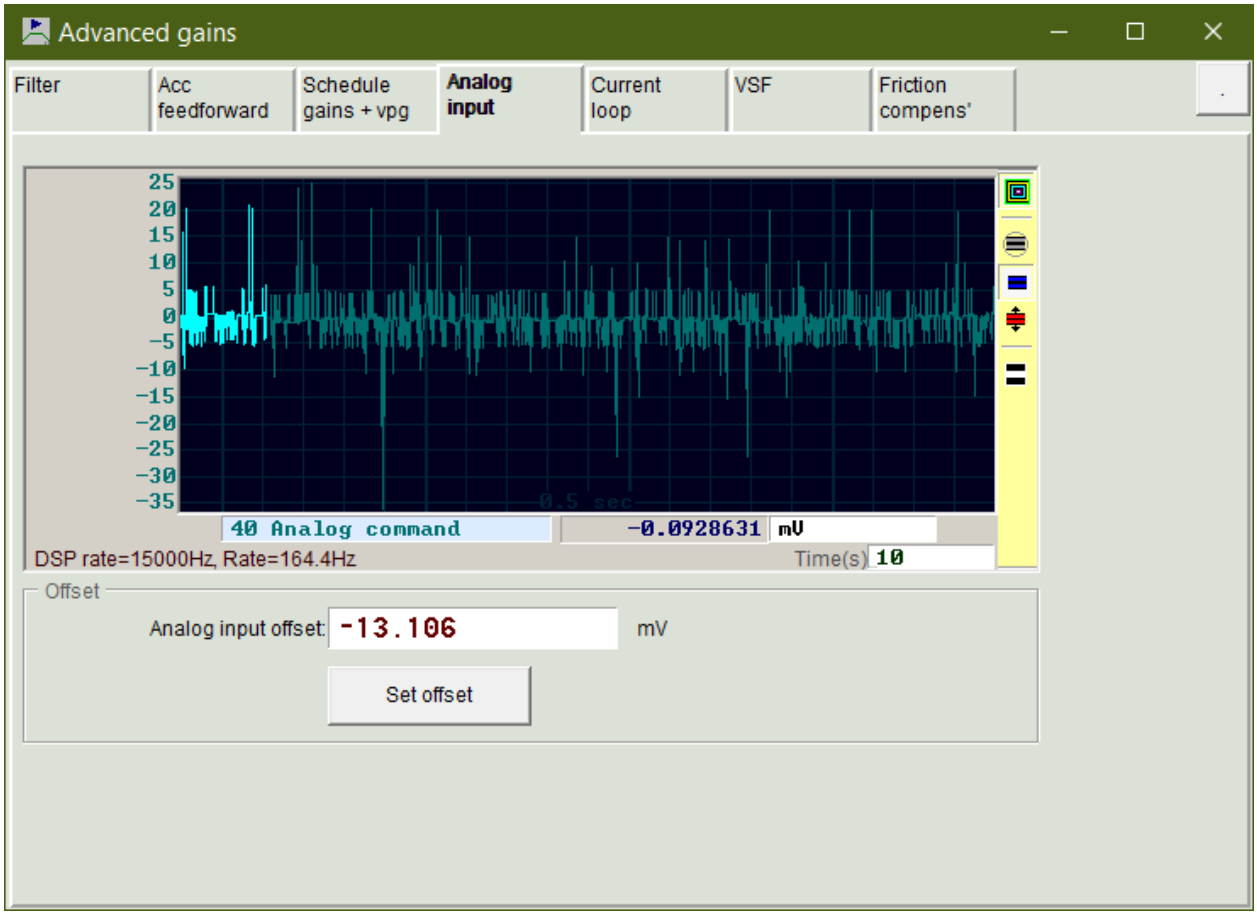


Figure C.18: This figure illustrates the bias correction of the analog signal.

of the CN2 channel is connected to the Speedgoat system. Next, the limit switch signal is read in the Simulink model, and once it is high, meaning the linear motor hits the limit switch, the hardware enable is deactivated. This will cut off all the power supply to the linear motor. Moreover, it will also stop the Simulink model from executing, thus stopping the experiment. The above process completes the I/O configuration of the linear motor drive.

Due to the EMI noise in the electrical system, the analog signal that controls the linear motor will contain some level of DC bias. In velocity mode, this DC bias constantly moves the pendulum cart to one side of the linear rail and thus should be avoided. By selecting

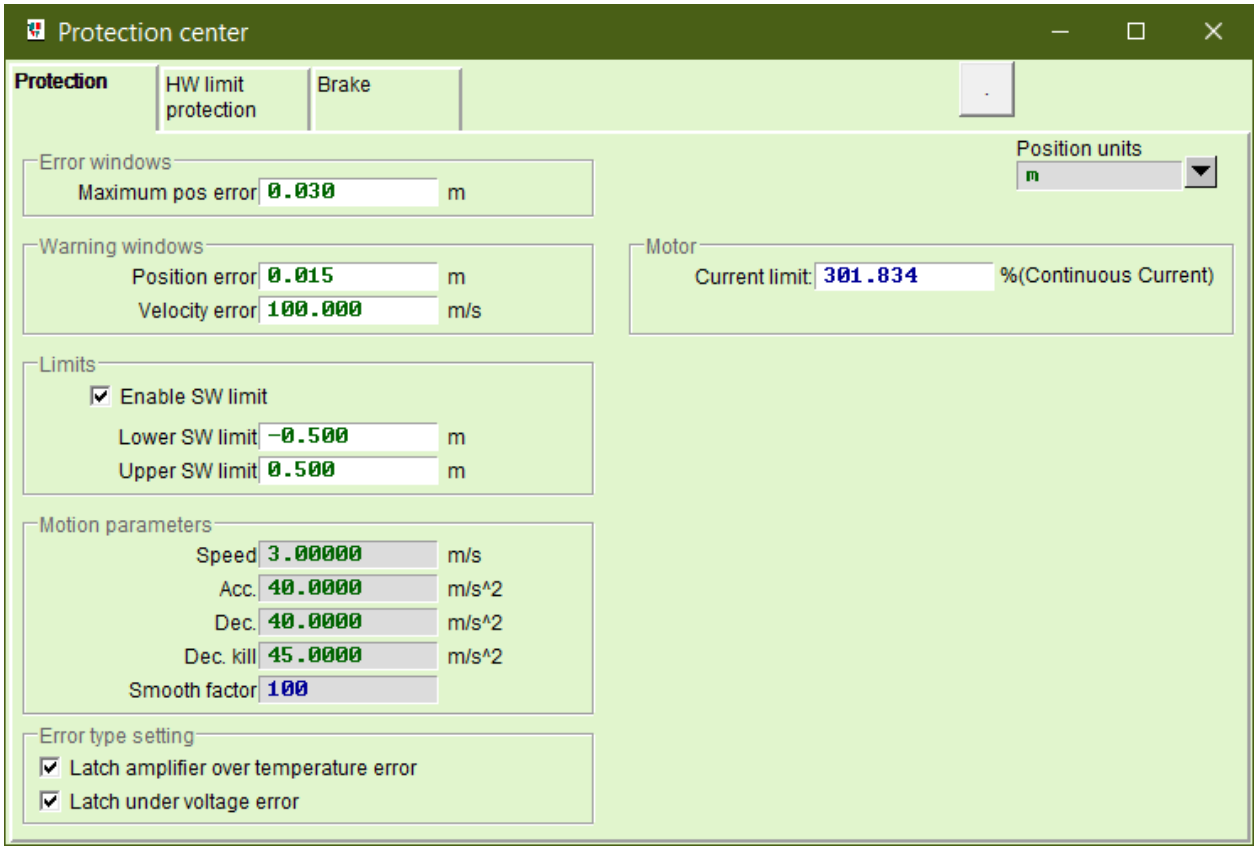


Figure C.19: This figure illustrates the software safety configuration by using Lightning software.

"Conf./Tune->Performance Center" to enter the performance center, the "Advanced Gain" button can be pressed, which will open up a window with multiple tabs. Selecting the "Analog Input" tab will result in Fig. C.18, where the users can press the "Set offset" button to minimize the effect of the analog signal's bias.

The safety mechanism is the final configuration that needs to be made in the Lightning software. To do so, we need to select "Conf./Tune->Protection Center" which will result in Fig. C.19, where one can set up the software position, velocity, and acceleration limit. Set up a narrow range for the limits and gradually increase it if the desired position, speed, or acceleration is out of range. This safety measure prevents unwanted high-speed movement

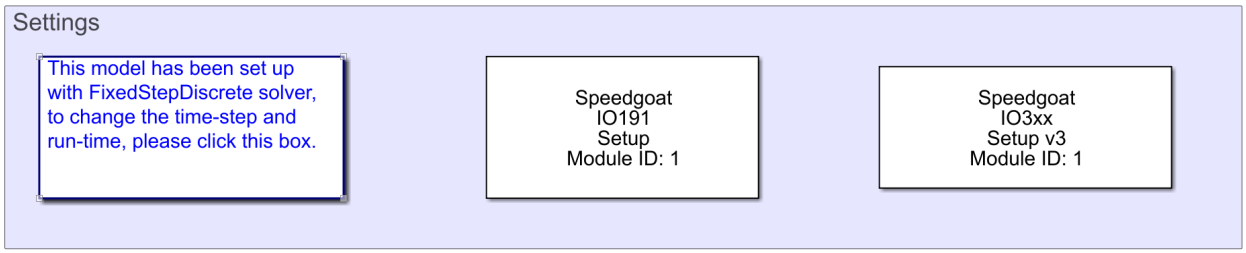


Figure C.20: This figure illustrates the two block sets used in the Simulink model that configures the Speedgoat IO-191 and Speedgoat IO-392 modules.

of the linear motor during the development phase of the controller.

The configurations of Lightning software in this section allow quick setup of the linear motor. For more details on using the Lightning software and its other functionalities, please refer to sections 4 and 5 of HIWIN D1 drive's user manual [450].

C.6.2 Software Set Up of the Simulink Model Details

The Simulink Real-Time is used to develop the controller used in the real-time pendulum experiments. The Simulink Real-Time and Speedgoat must be appropriately configured to read the sensors and output control signals. The first step of the configuration physically connects the Speeadgoat machine with the host computer using an Ethernet cable. This connection allows the user-programmed Simulink controller to be downloaded to the Speedgoat device and run in real-time. Moreover, this connection also enables the Speedgoat to send back the sensor signal at the run time and display it on the host computer using the "Data Inspector" function of the Simulink. Matlab and Speedgoat already have numerous tutorials on setting up the connection between the Speedgoat and the host computer. Thus we will skip them in this paper. The interested user should reference the documentation of the Speedgoat for configuration details.

Two block sets must be set up and used in the Simulink file to configure the I/O module IO-191 and FPGA module IO-392 for I/O functionalities and encoder reading. They are the

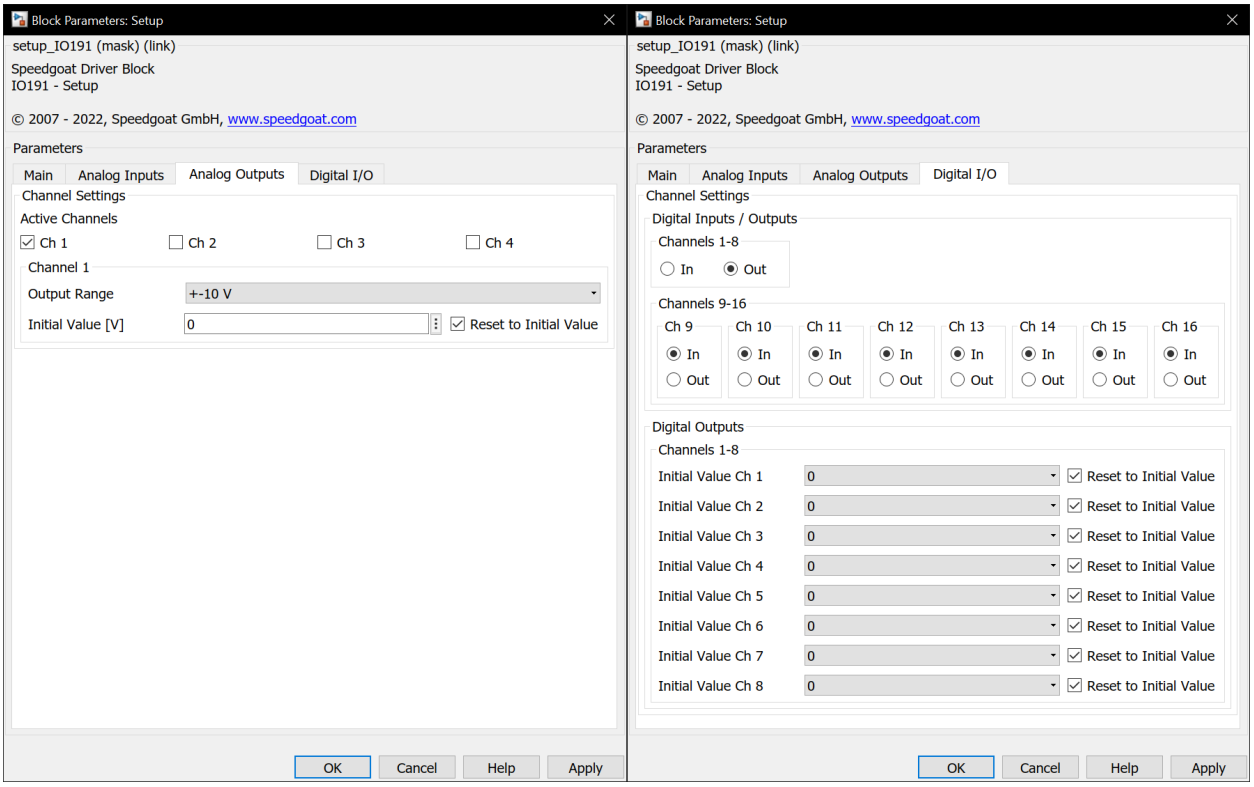


Figure C.21: This figure illustrates the configuration of the IO-191 module's Simulink block. One of the analog signal outputs is selected to control the velocity of the pendulum cart in velocity mode. Meanwhile, eight ports of the digital I/O block are chosen as input ports, and the rest are set to output ports.

"Speedgoat IO-191 Setup" block and "Speedgoat IO-392 Setup" block shown in Fig. C.20. Speedgoat provides those block sets when purchasing the Speedgoat machine through an active maintenance agreement. Due to the user agreement, the authors can not share those two block set installers. The interested reader should contact Speedgoat for purchase and installation guidance. To set up the I/O module, double click on the "Speedgoat IO-191 Setup" block and configure the "Analog Outputs" and "Digital I/Os" as Fig. C.21 shows. In the "Analog Outputs" tab, one of the channels is used as analog output with a range of $\pm 10V$. This analog output connects to the motor drive's analog input for the linear motor's speed control in velocity mode. In the digital I/O tab, the first eight channels are

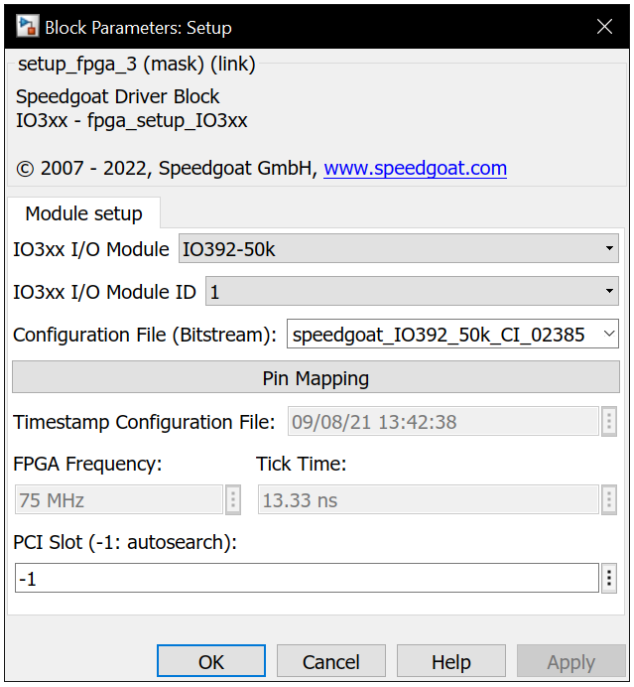


Figure C.22: This figure illustrates the configuration of the IO-392 module's Simulink block. The bitstream file provided by Speedgoat is needed to set up the IO-392 block to read encoder data properly.

used as the digital output ports, while the rest are configured as digital input ports. Three of the digital outputs are used to control the "Hardware Enable," "Switch to Secondary Mode," and "Start Homing" functions of the motor drive, as Fig. C.17 shows. Two of the digital inputs are used to read the signals of limit switches. Table. C.2 shows the detailed connection. The above process completes the IO-191 module configuration for real-time digital I/O functionalities.

To set up the IO-392 block set, double click on the "Speedgoat IO3XX Setup v3" block and select the "IO-392-50k" for "IO3xx I/O Module" drop-down list. Moreover, the configuration file (Bitstream) "speedgoat_IO392_50k_CI_02385" is used as Fig. C.22 shows. This bitstream file needs to be in the same folder as the Simulink file. Moreover, this file can be thought of as the driver of the FPGA module. It is purchased separately and is provided by the Speedgoat according to the user's need. The interested reader should

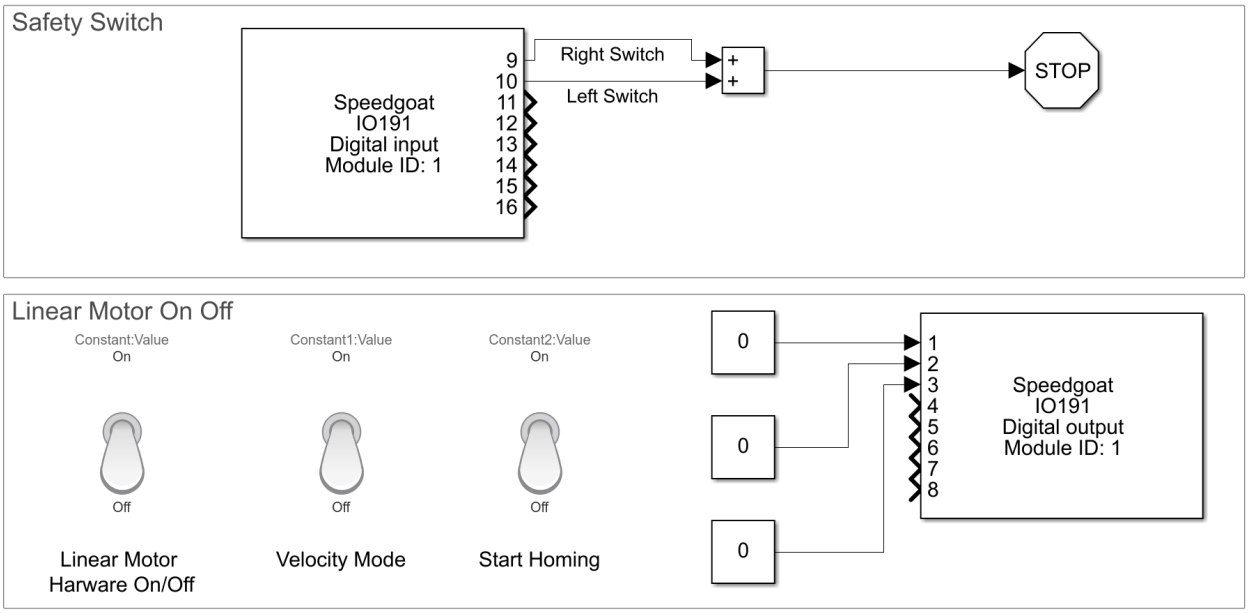


Figure C.23: This figure shows enabling and disabling of the linear motor in Simulink. It also indicates how to switch the operating mode of the linear motor. By turning on the "Start Homing" toggle, the linear motor can start homing functionality in the stand-alone mode. The left and right limit switch signal is read and added up to stop the experiments if the limit switch is triggered. If one of the signals is turned to high, it will activate the "Stop Simulation" button of the Simulink, and the entire experiment is terminated.

contact Speedgoat for a quote.

Fig. C.23 shows the start and homing functionalities of the linear motor. A toggle switch is used in the simulation file to hardware start the linear motor. This toggle switch controls the "Digital I/O 01" output of the IO-191 module. Thus, when the toggle is switched on, meaning the production of pin 1b is high, the D1 drive will receive an "activate" signal on the third pin of the CN2, and thus hardware activates the linear motor as configured in Fig. C.17. When hardware-enabled, the default operating mode the linear motor is in is the stand-alone mode. Users can switch on the "Velocity Mode" toggle to enter the velocity mode. This action will send out a high signal on the 2b pin of the Speedgoat machine, which is connected to the fourth pin of the D1 drive's CN2 channel, which will then set the motor drive to the velocity mode as Fig. C.17 configures. When turned on the first toggle

and turned off the second toggle, the D1 drive will be in stand-alone mode. This action sets the "Digital I/O 03" to high so that the motor drive receives the "Start Homing" command and positions the linear motor to the middle point of the left and right limit switches. In Fig. C.23, the "IO-191 Digital Input" block receives the signals of limit switches. In real-time execution, those signals are constantly checked. When the limit switch is not triggered, the reading is low and vice versa. Thus, by summing them up together and connecting the summation to the "Stop Simulation" button, the simulation can be terminated when any limit switches get activated. The termination of the run file will set the output of all the ports to the low state (default), setting the linear motor to hardware disabled mode. This completes the setting up of the digital I/O functions in the Simulink file.

The desired analog voltage output of the IO-191 module needs to be supplied as Fig. C.24 shows to control the linear motor's motion. A saturation block limits the voltage input range in $\pm 10V$ to avoid overflow issues. The desired voltage output is calculated using

$$Volts = \frac{v}{K_a}, \tag{C.1}$$

where $Volts$ is the desired analog voltage output, v is the desired velocity of the linear motor, and K_a is the scaling factor between the two. In our set up, we used $K_a = 0.3$ as Fig. C.16 shows. The analog output of the IO-191 module connects the analog input of the D1 drive so that the motor drive can receive the velocity command. The analog ground of the IO-191 and motor drive is connected as well. Table. C.2 shows the detailed wiring, thus completing the controlling of the linear motor velocity.

Fig. C.24 also shows the reading of the encoder signal. The "IO-392-50K QAD V3" block is used to read the encoder signal. This block is purchased along with the IO-392 module. To set up this block, one needs to double click on it and then set the correct channel vector number of the encoders one wants to read in the "Module setup" tab. As Table. C.1 shows, the linear motor's encoder is wired to the fourth encoder channel. Thus, "[4]" should be entered in the "Channel vector" blank of the encoder block. Next, the reading mode of the

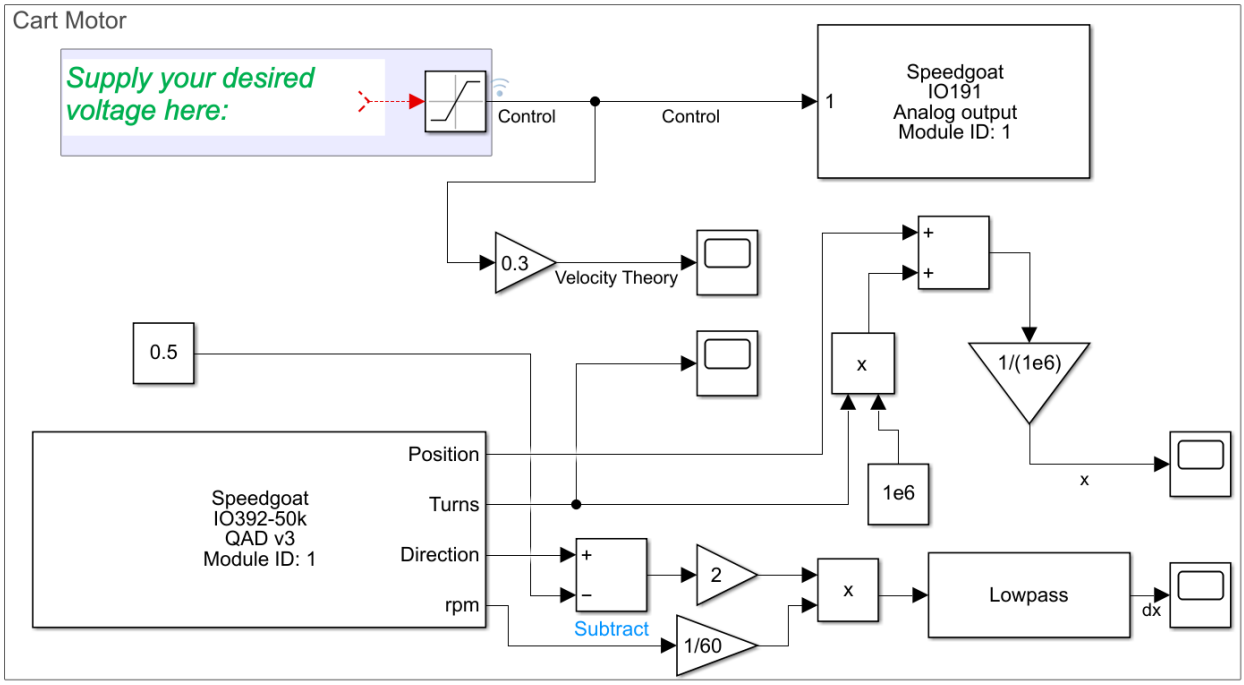


Figure C.24: This figure illustrates the configuration of the pendulum cart in the Simulink model. To control the linear motor's velocity, the user should input desired analog voltage to the "Analog output" block. The output of the analog signal will be received by the D1 drive to control the linear motor to the user-defined speed. The "QAD V3" block is used to read the position and velocity of the linear motor. Moreover, some post-processing is needed to read the correct value of the linear motor's speed and velocity.

encoder is configured in the "Quadrature decoder" tab by setting the "Operating mode" as "Quadrature decoder" and "Sampling" as "$4\times$". The "Latch mode" is set to "Disabled" while "Interpolate inner position" is turned on. The correct "Steps per revolution vector" needs to be configured, which defines the counting range of the encoder counter. In the case of the linear motor's encoder, its resolution is $1\mu m$, which means there are 10^6 counts per meter. Thus, this value is entered as 10^6 (or 4×10^6). The default reading behavior of the encoder block wraps around to zero if the encoder reading value is larger than the value entered in the "Steps per revolution vector" blank. Some post-processing needs to be done to avoid this and achieve continuous reading that does not warp around. This is achieved by first selecting the "Show position," "Show turns output," "Show speed output,"

and "Show direction flag output" in the "Input and output port configuration" tab of the block. Fig. C.24 shows the resulted output ports of the encoder reader block. The "Turns" output will change accordingly whenever the encoder disk or linear motor performs a complete revolution. If the steps increase and a full revolution is made, then the "Turns" increase by one and vice versa. Thus, the total number of steps reflecting the absolute position of the linear motor or encoder disk can be calculate using

$$Absolute\ Steps = Steps + Turns \times (Steps\ per\ Revolution), \tag{C.2}$$

where the "Steps" variable is the output of the "Position" port of the encoder reader block, "Turns" variable is the output of the "Turns" port. To transfer the $Absolute\ Steps$ into the "$AbsolutePosition$" of the linear motor, the "$Steps\ per\ Revolution$" is used to divide the $Absolute\ Steps$, which will result the absolute position value of the linear motor (in m). This can be described as

$$Absolute\ Position = \frac{Absolute\ Steps}{Steps\ per\ Revolution}, \tag{C.3}$$

which completes the task of position reading of the linear motor. Sometimes, the velocity of the linear motor is also wanted, and it can be calculated using the "rpm" and "Direction" output port. The first thing to do is transfer the "rpm" into rounds per second. Dividing the output of the "rpm" port by 60 achieves this task. However, the output of the "rpm" port is the absolute value of the velocity. Thus, to include directional information, the "Direction" port's output is transformed into 1 or -1, where 1 represents positive velocity while -1 represents the negative velocity. Then, it is multiplied with rounds per second to get the directional speed of the linear motor. The user can add a "Lowpass" filter block to denoise the encoder's velocity measurement, which completes the reading of the linear motor encoder.

Fig. C.25 shows the user can configure a similar setup to read the angular position

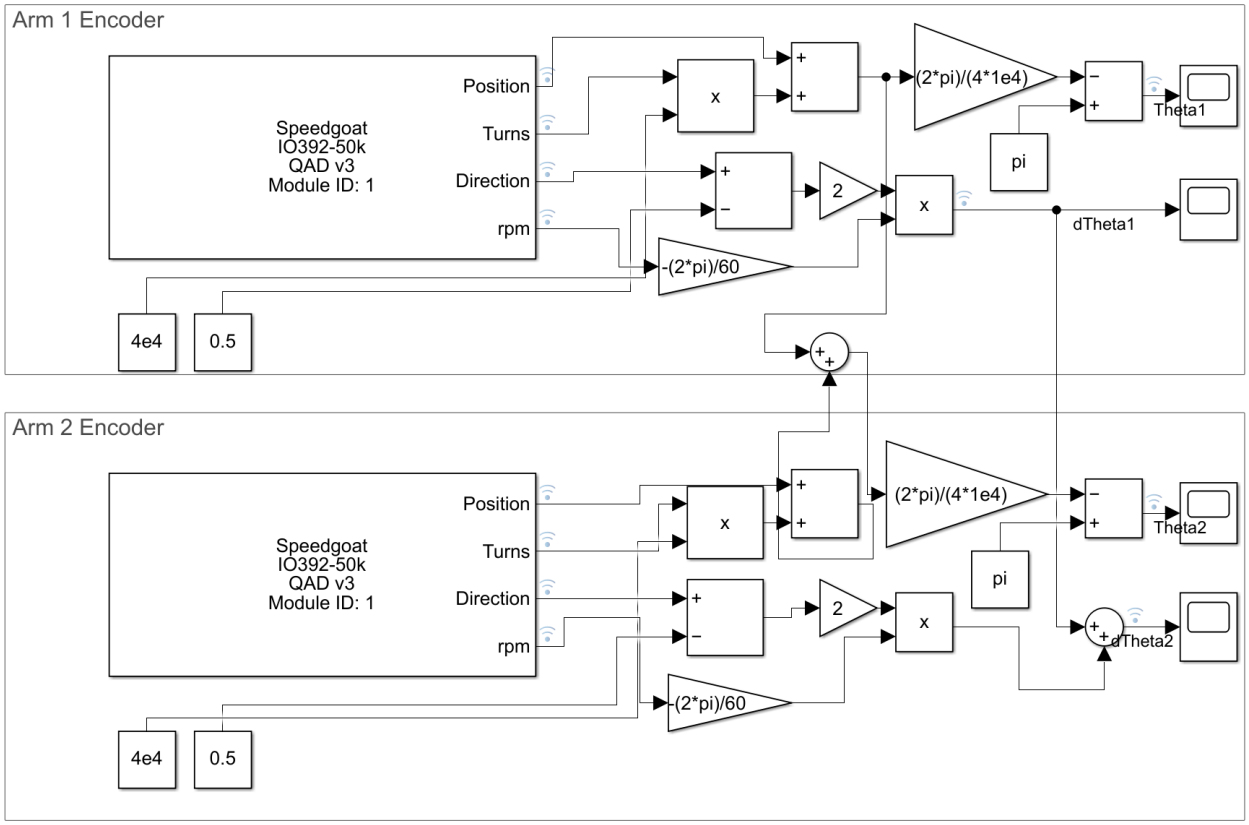


Figure C.25: This figure illustrates how to read the angular position of the double pendulum shown in Fig. 6.1. Conversion in Eq. (C.5) and Eq. (C.6) is needed to read the angular position of the second pendulum arm defined in Fig. 6.1. A similar configuration of the encoder block is performed as we did in Fig. C.24 with a different value for the "Steps per revolution vector" blank.

of the pendulum arm, and we use double pendulum encoder reading as an example to illustrate this. First, as we did in the linear motor encoder reading configuration, the "Channel vector" value of two blocks is defined. As Table. C.1 indicates, the first arm's encoder channel vector should be "[1]," the second arm's channel vector should be "[2]", and when using the triple pendulum arms, the third encoder's channel vector should be "[3]". Next, the "Operating mode" is selected as "Quadrature decoder," and "Sampling" is selected as "$4\times$". Similarly, the "Latch mode" is set to "Disabled" while "Interpolate inner position" is turned on. The "Steps per revolution vector" is set to 4×10^4. To read the

absolute rotational angle of the pendulum arm in radius, the following equation is used

$$Absolute\ Rotational\ Angle = \frac{Absolute\ Steps}{Steps\ per\ Revolution} \times 2\pi. \tag{C.4}$$

Moreover, the user can calculate the angular speed of the pendulum arm in radius by multiplying the output of the "rpm" port with $1/60$ to transfer it to the rounds per second. Next, it can be multiplied by $\pm 2\pi$ to transfer the velocity into rad/s. The choice of plus or minus sign determines whether the pendulum arm's clock-wise rotation increases or decreases the encoder reading. In our case, we define the clock-wise rotation of the pendulum arm increases the value of the encoder reading. Finally, it is vital to note that the reading of the second pendulum arm does not correspond to the θ_2 value shown in Fig. 6.1. Define the encoder reading of the first, second, and third arm as α, β and γ. Then the relationship between the encoder reading and pendulum arm's angle θ_1, θ_2, and θ_3 can be written as

$$\theta_1 = \alpha, \tag{C.5a}$$

$$\theta_2 = \alpha + \beta, \tag{C.5b}$$

$$\theta_3 = \alpha + \beta + \gamma. \tag{C.5c}$$

Thus, the angular speed of the pendulum arm can be written as

$$\dot{\theta}_1 = \dot{\alpha}, \tag{C.6a}$$

$$\dot{\theta}_2 = \dot{\alpha} + \dot{\beta}, \tag{C.6b}$$

$$\dot{\theta}_3 = \dot{\alpha} + \dot{\beta} + \dot{\gamma}. \tag{C.6c}$$

We provide a template file that achieves all the encoder reading and linear motor control tasks[3]. The reader can directly modify this file to include their controller without perform-

[3]Code availalbe at https://github.com/dynamicslab/MultiArm-Pendulum

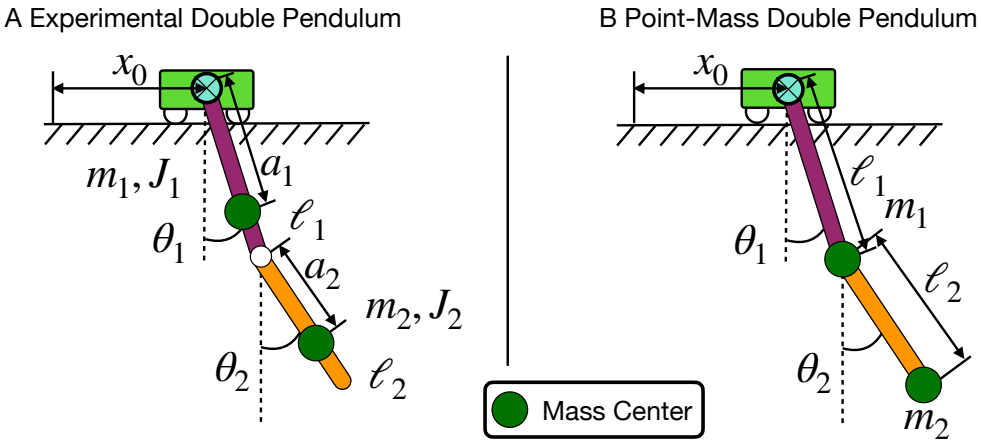


Figure C.26: This figure illustrates the difference between the point-mass model of the double pendulum (left) and the more realistic experimental double pendulum (right) studied herein. The point-mass model ignores the mass of each arm of the pendulum, while the experimental model considers both the inertial (J_i) and mass (m_i) of the pendulum arms.

ing the encoder reading configuration. This conversion is also shown in Fig. C.25, which completes the setup of the Simulink configuration.

C.7 Step by Step Operation Details

C.7.1 Pre-Experiment Preparations

Before starting the experiments, the following operations are needed:

1. Put on safety glasses.

2. Determine the overall sampling frequency of the setup, control input range, cart velocity range, and cart position range. These parameters can help determine whether the purposed experiments is physically realizable. If the desired cart velocity and acceleration range is higher than the limit introduced in Fig. C.19, change the limits in Fig. C.19 accordingly.

3. Develop the controller Simulink file, and make sure it compiles correctly. Rewire the connection of the IO-191 module if the custom I/O functionalities are needed.

4. Double check the wiring between the Speedgoat terminal block and pendulum arm encoder and linear motor drive.

5. Check the connection between the slip-ring and slip-ring brush. Make sure the slip-ring and slip-ring brush are contacting each other.

6. Connect the host computer and Speedgoat machine using an Ethernet cable. Turn on the Speedgoat machine and setup the Simulink Real-Time to establish communication between the host computer and Speedgoat machine. This step and those prior to it complete the connection between the pendulum arm and Speedgoat machine.

7. Check all the wires that connect the linear motor's electrical parts mentioned in Fig. 6.12. Make sure all cables are connected properly according to the wiring diagram shown in Fig. C.11.

8. Release the emergency stop button and connect the circuit breaker.

9. Plug in the power cord of the emergency stop button module. This power cord is used to supply power to the linear motor, motor drive, and all other components in the electrical parts of the linear motor.

10. Press the green button of the emergency stop button module to supply AC single phase power to the electrical box of the system. Wait a moment to observe if any shortage has occurred. If so, press the emergency stop button to cut off the power supply. If not, proceed to the next step.

11. Press the green momentary switch to self-lock the magnetic contactor. This will provide the power supply to the line reactor and DC power supplied module. Wait

a moment to observe if any shortage has occurred. If so, press the emergency stop button to cut off the power supply. If not, proceed to the next step.

12. Set the toggle switch of the electrical box to on. This provides power to the motor drive, and in turn the motor drive will provide power to the linear motor. Wait a moment to observe if any shortage has occurred. If so, press the emergency stop button to cut off the power supply. If not, proceed to the next step.

13. Remove any and all obstacles from the motor rail.

14. Execute the Lightning software and connect the D1 motor drive to the host computer. Make sure all the parameters shown in the Lightning software are correct.

15. Open the Simulink model with developed controller. Compile it and download it to the Speedgoat machine.

16. Execute the compiled controller, start the desired experiments.

C.7.2 Peri-Experiment Operations and Precautions

One should adhere to the following operations and precautions below when performing experiments:

1. Never get close to the moving pendulum cart just in case any mistakes in the designed controller file make the linear motor perform undesired motion and injure the operator.

2. The operator should stand on the front or back side of the experimental setup. Never stand on the side of the linear motor as one may be struck by a pendulum arm.

3. The operator should closely monitor the behavior of the experimental setup. If for any reasons there is an electical shortage, smoke, heat, or unwanted noise, immediately

press the emergency stop button to cut off the power supply to the linear motor to avoid further damage.

4. The operator should closely monitor the surroundings of the experimental setup. If any person, animal, or unwanted object enters the experimental area the experiment should be stopped immediately to avoid injury and/or prevent the damage of the experimental setup.

5. The operator should continually check the data being collected from the experiment. If an unwanted measurement drift is apparent in the plotted data, immediate stop the experiment by pressing the stop simulation button of the Simulink file.

C.7.3 Post-Experiment Operations

After an experiment, the following operation should be performed:

1. Check all wiring related to the electrical component of the pendulum setup. Reconnect any wires that may have become disconnected during the experiment.

2. Check for damage to the system hardware.

3. Go back to the pre-experiment operations to perform another experiment or follow the proceeding steps to turn off the system.

4. Close the Lightning software to cut off the communication between the D1 drive and host computer.

5. Switch the toggle switch to off to cut off the power to the motor drive.

6. Press the red momentary switch to cut off the power to the DC power supply and line reactor.

7. Press the off button of the emergency stop button module. This will cut off all power supply to the linear motor. Then unplug the power cord.

8. Save the experimental files and data to a secured location. After saving close Matlab and Simulink.

9. Turn off the Speedgoat machine. This will cut off the power supply to the pendulum arm encoder.

10. Turn off the host computer.

With the above post-experiment operations, all of the power supplied to the experimental system is now cut off.

C.8 Safety Mechanism and Notes Details

The following electrical safety mechanisms should further be implemented:

1. Use the circuit breaker protect the motor from over current failure.

2. Install the limit switch on the linear motor to prevent the out of range motion of the linear motor.

3. Use the regenerative resistors to absorb extra kinetic energy during the breaking of linear motor.

4. Use the noise filter to reduce the ground current protecting the appliances that share the same ground.

5. The noise filter, shielded cable, edge filter, and EMI cores are used to reduce the side effect of EMI noise. This in turn provides a cleaner signal and prevents the possible damage caused by faulty measurements signal.

6. Obtain better control of the electrical system using the emergency stop button, momentary switches, while allowing for rapid disconnection from the power supply, thus improving safety in the case of any electrical shortage.

7. Use an electrical box to prevent potential mechanical damage to the electrical components.

8. Properly ground electrical components to avoid a shock to the operator.

9. Use the linear motor's temperature sensor to prevent overheating and over current failure.

The Lighting and Simulink software should be modified as follows to guarantee user safety:

1. Use the Lightning software to internally restrict the range of linear motor's position, velocity, and acceleration.

2. Use the Lightning software to stop the linear motor when the limit switch is triggered.

3. Program the Simulink model to limit the analog output, which restricts the velocity range of the linear motor.

4. Use the Simulink model to stop the experiment when the limit switch is triggered.

In addition to the above safety procedures, one should also use the following points to ensure the safety of the operator and surrounding personnel.

1. Always check the wire connections for damage or disconnections before starting the experimentto avoid an electrical shortage and fire.

2. Clear the linear motor rail of obstacles before supplying power to the motor.

3. Do not touch the linear motor rail by hand. This could potentially rust the motor rail.

4. Apply general purpose grease every month to protect the motor rails from rusting.

5. Never touch the linear motor while it is in operation.

6. Never perform any wiring of electrical components with the power supply on.

7. Keep a safe distance from the system while it is operating. We suggest the operator and all surrounding personnel should be at least one meter from the experimental setup while performing experiments.

8. Do not stand to the side of the pendulum arms as this increases the risk of being hit by a pendulum arm.

9. Do not expose the electrical components to moisture or liquid as they are not water-proof. Furthermore, do not operator the equipment if hands are wet.

10. Wear safety glasses for all construction and operation steps for the experiment.

C.9 Parameter Estimation Details

This Appendix. gives the details of parameters estimation process shown in Table. C.4. In this Appendix., we will first use the EOM of the double pendulum on the cart as an example to explain why it is desired to use the acceleration of the pendulum cart as our control input. Then, we will show more details on the parameter estimation process we used to identify the parameters in Table. C.4. Since the model derivation task of single, double and triple pendulum are standard text book example, we refer the reader to check [7, 428] for more details on deriving the EOM of pendulum arm on the cart.

Table C.4: Comparison of the estimated parameters and CAD model estimated parameters of the single, double, and triple pendulum. The length l_1, l_2 and l_3 are not estimated for the single, double, and triple pendulum respectively, since they are not present in the corresponding EOM. It is interesting to note that the estimated values of the local gravity constant g are different for all three pendulums. This is caused by the fact the g is an optimization parameter during the parameter estimation and is allowed to vary to best fit the data. Units: mass kg, length m, inertia kgm^2, and local gravity constant m/s^2.

Pendulum Type	Single		Double		Triple	
Source / Vars	CAD	Estimated Parameters	CAD	Estimated Parameters	CAD	Estimated Parameters
m_1	0.1183	0.1476	0.1199	0.0938	0.11	0.2582
m_2			0.1183	0.1376	0.12	0.2794
m_3					0.1	0.1186
l_1	0.17272		0.17272	0.1727	0.1727	0.1728
l_2			0.2286		0.2286	0.2287
l_3					0.2413	
a_1	0.08638	0.1478	0.08755	0.1086	0.08755	0.16
a_2			0.1256	0.1168	0.127	0.2029
a_3					0.12	0.1837
J_1	0.001578	1.0912×10^{-4}	0.001627	10^{-4}	0.001578	1×10^{-4}
J_2			0.002908	10^{-4}	0.0029	3.1416×10^{-4}
J_3					0.002157	1.6874×10^{-4}
ε_1		2.2394×10^{-4}		10^{-5}		0.0015
ε_2				10^{-5}		2.5958×10^{-4}
ε_3						1×10^{-5}
g	9.8083	9.81001	9.8083	9.808	9.8083	9.8083

C.9.1 Equations of Motion

This section shows why it is desired to use the acceleration of the pendulum cart as control input and we illustrate this using double pendulum as an example. Let x_0 be the position of pendulum cart and v_{x_0} its velocity, and the rotational angle of pendulum arm as θ_1 and θ_2 respectively. Following the modeling process shown in [7], the second derivative of the double pendulum's rotational angle can be written as $\ddot{\theta}_1$ and $\ddot{\theta}_2$ as Eq. (7.18).

In Eq. (7.19) we use the convention that A_{ij}, B_{ij} are the coefficients of the energy preserved Hamiltonian system for the $\ddot{\theta}_1$ and $\ddot{\theta}_2$ equations, respectively, while the subscripts denote the coefficients on the θ_1 and θ_2 terms inside the sine function they are multiplied

against. The F_{ij} are the $(\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2)$-dependent friction terms in $\ddot{\theta}_i$ equation with friction coefficient ε_j. Finally, the C_i terms correspond to the influence of the cart velocity, $\dot{v}_{x_0}$, for the $\ddot{\theta}_i$ equation and $D(\theta_1 - \theta_2)$ represents the denominator function. By writing the equation of the of the experimental double pendulum in the form of Eq. (7.18), one can see the acceleration of the pendulum cart $\dot{v}_{x_0}$ can be nicely used as control input of the system. By **assuming** perfect velocity control can be provided by the motor drive, the mass of the pendulum cart M can be avoided in the equation of motion of the pendulum cart.

In experiments with the physical double pendulum on a cart the velocity of the pendulum cart is controlled by the servo drive by applying an analog voltage input as we mentioned in Sec. 6.7.1. This voltage input is calculated by first taking the integral of the acceleration of the pendulum cart $\dot{v}_{x_0}$. The integration will result the desired velocity of the pendulum cart v_{x_0}. Then the desired velocity is scaled to the corresponding voltage using the scaling factor shown in Fig. C.16. Thus, in the experiments, the acceleration of the double pendulum is chosen as the control input of the system. To account for this control input, we write out the complete first order ODE of the pendulum on the cart system using acceleration of the cart as control input, denoted u. The final result is the ODE shown in Eq. (7.20).

System 7.20 therefore represents the full equations of motion for the double pendulum on a cart that we investigate for the parameter estimation task. The equations of motions shown in Eq. (7.20) made various assumptions. For example, it did not consider the the air drag of the pendulum arm, it did not consider the effect of rolling bearing balls during the rotational movement as shown in [446], it did not consider the time-lag between the control signal and movement of the linear motor, it did not consider the oscillation of the pendulum, it did not consider the fact that linear motor might not be moving in a horizontal line, etc. However, it does a pretty good job on capturing most of the system dynamics and is the most widely used equation of motion for the experimental double pendulum on the cart system. If the user consider more factors, a more accurate and complicated model of the double pendulum on the cart can be derived. The same procedures shown above

can also be used to derive the equation of motion of the single and triple pendulum on the cart. A good reference on deriving the equation of motion of the triple pendulum can be seen in [428].

C.9.2 Parameter Estimation

Having now derived the equations of motion for the experimental double pendulum on the cart, we now seek to identify the values of the numerous parameters to arise in it. Our goal is to align the parameters of the model (7.20) with those of our constructed double pendulum on a cart. Of course, the parameters of the physical model are uncertain and we expect a difference between the actual and theoretical parameter values due to imperfect manufacturing, installation tolerance, etc. In this subsection, we will present the method we have used for parameter estimation of the double pendulum. The same approach is also used to estimate the parameters of the double and triple pendulum.

To estimate the parameters of the double pendulum we fix the position of the cart. The result will be that $u = 0$ in (7.20), thus reducing the ODE to one in the four variables $(\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2)$, given by

$$\begin{aligned}
\dot{\theta}_1 &= \omega_1, \\
\dot{\theta}_2 &= \omega_2, \\
\dot{\omega}_1 &= \frac{A_{10}\sin(\theta_1) + A_{11}\sin(\theta_1 - \theta_2) + A_{12}\sin(\theta_1 - 2\theta_2) + A_{22}\sin(2\theta_1 - 2\theta_2) + \varepsilon_1 F_{11} + \varepsilon_2 F_{12}}{D(\theta_1 - \theta_2)}, \\
\dot{\omega}_2 &= \frac{B_{01}\sin(\theta_2) + B_{11}\sin(\theta_1 - \theta_2) + B_{21}\sin(2\theta_1 - \theta_2) + B_{22}\sin(2\theta_1 - 2\theta_2) + \varepsilon_1 F_{21} + \varepsilon_2 F_{22}}{D(\theta_1 - \theta_2)}
\end{aligned} \tag{C.7}$$

Next, the experimental data of the pendulum arms is collected by swinging them from an arbitrary initial condition. The relative angles of pendulum arms with respect to its joint is measured using the encoders, with 10000 CPR accuracy in quadrature mode, which results in 40000 PPR (pulse per revolution). Suppose that we have collected $n \geq 1$ measurements

of the values of $(\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2)$, each over $L \geq 1$ time values. We arrange each measure into the data matrices $\mathbf{Y}_i \in \mathbb{R}^{4\times L}$ for each $i = 1, \ldots, n$ and denote $\hat{\mathbf{Y}}_i(p) \in \mathbf{R}^{4\times L}$ the data generated by simulating (C.7) with the same initial conditions for each $i = 1, \ldots, n$ and a given choice of the parameter values

$$p = [m_1,\ m_2,\ a_1,\ a_2,\ \ell_1,\ J_1,\ J_2,\ \varepsilon_1,\ \varepsilon_2,\ g]. \tag{C.8}$$

Then, the goal of our parameter estimation is to solve the optimization problem

$$\min_p \sum_{i=1}^{n} \|\mathbf{Y}_i - \hat{\mathbf{Y}}_i(p)\|_2. \tag{C.9}$$

Of course, a (global) minimum of the above error identifies a choice of parameters for which the trajectories of the simulated double pendulum using (C.7) most closely aligns with that of the physical model.

To perform our parameter estimation we have used $n = 26$ different measurements, all sampled with sampling rate of $1kHz$. Each $\mathbf{Y}_i$ consists of $L = 2667$ consecutive data points. We solve the optimization problem (C.9) using MATLAB's optimization toolbox. Specifically, we employ the *particleswarm* algorithm to achieve the global minimum. To better assist the convergence to the global optimum, we initialize with upper and lower bound constraints on the parameter values. These bounding values and their units are given component-wise by

$$\begin{aligned} p_l &= [0.09\text{kg},\ 0.09\text{kg},\ 0.09\text{m},\ 0.10\text{m},\ 0.172719\text{m},\ 10^{-4}\text{kgm}^2,\ 10^{-4}\text{kgm}^2,\ 10^{-5},\ 10^{-5},\ 9.808\text{m/s}^2] \\ p_u &= [0.14\text{kg},\ 0.14\text{kg},\ 0.16\text{m},\ 0.22\text{m},\ 0.172721\text{m},\ 0.002\text{kgm}^2,\ 0.002\text{kgm}^2,\ 10^{-3},\ 10^{-3},\ 9.809\text{m/s}^2]. \end{aligned} \tag{C.10}$$

As is apparent from (C.10), we begin the optimization process by being more certain of some parameter values than the other from the aid of simple measurement devices such as electric scales. In contrast, there is not a good way to directly measure the the inertial

and friction coefficients of the pendulum arms, and so we have provided a larger variance in our upper and lower bounds. However, the CAD models of the pendulum arms can be used to have a rough guess of the inertial parameters. The result of the parameter estimation process using the minimization scheme (C.9) is

$$\begin{aligned} p_* = [&0.0938\text{kg},\ 0.1376\text{kg},\ 0.1086\text{m},\ 0.1168\text{m},\ 0.1727\text{m}, \\ &10^{-4}\text{kgm}^2,\ 10^{-4}\text{kgm}^2,\ 10^{-5},\ 10^{-5},\ 9.808\text{m/s}^2]. \end{aligned} \tag{C.11}$$

It should be noted that the parameters we identified in Eq. (C.11) is **NOT** the real physical parameters of the experimental double pendulum. Rather, it should be think of as a set of optimization variables that optimize the minimization scheme (C.9) while at the same time is numerically **close to** the actual experimental parameters. This is due to the fact that model we used does not capture all the dynamics of the pendulum arm and the data we collected is not perfectly clean[4]. Thus, there's rooms for the optimization algorithms to fine tune the optimization variables to compensate for unconsidered factors during the modeling. This is why the global optimization algorithm is used to find out a set of parameters that best compensate those unmodeled factors. Another important thing to note is, the identified model parameters provides a good simulated forward prediction performance if the dynamics of the double pendulum under prediction is similar to those used in parameters estimation. In other words, using data set that covers different parts of the double pendulum's phase space will result various optimal solution for Eq. (C.11). This happens quite frequently when the control input is applied to the pendulum on the cart system, where often times an extra step of parameter tuning is needed to compensate the unmodeled behavior of the pendulum as shown in [7, 428, 436]. Another interesting thing we noticed is that the parameters estimated using a higher energy level of double

[4]The estimated angular velocity of the pendulum arm deviates from the true velocity. This makes the forward simulation of Eq. C.7 accumulates error due to the chaotic behavior of the double pendulum. This deviation of estimated velocity of the pendulum arm makes the estimated parameters deviates from the true physical value as well.

pendulum's motion requires a longer time to converge to the optimal solution. Moreover, it might not provide a good forward prediction performance for the low energy motions. We suspect the increase of the training time is due to the fact that the higher energy level motion is more sensitive to the change of parameters. As for the worse prediction performance on the low energy data, we believe it is caused by too much tuning been made in the high energy level data set to compensate for the unmodeled factors, since the unmodeled factors plays more role in the high energy data set. This is why we included $n = 26$ different measurements, including low, medium, and high energy level motion of the pendulum to get a balance performance on prediction performance. The parameter estimation task of the single and triple pendulum is similar to the one for double pendulum and is omitted here. Instead, the estimated parameter is directly given in Table. C.4.

C.10 Bill of Materials

C.10.1 Bill of Materials for Building Pendulum Arm

All the materials needed to build the pendulum arm shown in Fig. 6.4 can be seen in Table. C.5. The price of some of the parts are not shown and the readers need to request a quote from the company to get its price. Other parts such as screws are sold in pack, for those parts the price for one pack is listed and is indicated by "*" symbol. Besides the materials mentioned in Table. C.5, we also recommend ordering 1008-1010 carbon steel ring shim set (1/4" ID, McMaster 3088A928) to accommodate for the manufacturing error if needed.

Table C.5: Materials needed to build the pendulum arm shown in Fig. 6.4.

Designator	**Component**	**Number**	**Total cost-currency**	**Source of materials**	**Material type**
Arm1Shaft (1)	McMaster 1346K17	1	9.61	McMaster	1566 Carbon Steel

Table C.5 continued from previous page

Designator	Component	Number	Total cost-currency	Source of materials	Material type
SlipRing 8 Channel (2)	Moog AC2690-8	1	NA	Moog	Gold and Others
Arm1 (3)	McMaster 8975K224	1	27.4	McMaster	Aluminum
91251A077 (4)	McMaster 91251A077	16	8.74*	McMaster	Black-Oxide Alloy Steel
WireClip (5)	3D Printed	4	NA	NA	PLA
EncoderShim (6)	3D Printed	2	NA	NA	PLA
Encoder (7)	US Digital EM2-2-10000-I	2	87.02	US Digital	Others
91251A081 (8)	McMaster 91251A081	4	9.01*	McMaster	Black-Oxide Alloy Steel
BrushShim (9)	3D Printed	1	NA	NA	PLA
BrushBlock (10)	Moog AC259-5	1	NA	Moog	Gold and Others
3088A449 (11)	McMaster 3088A449	2	7.13*	McMaster	Steel
R188 [5] (12)	R188 Hybrid Ceramic Bearing	4	105.60	Ortech	Ceramic, Steel, and Others
Arm2Shaft (13)	McMaster 1346K11	1	8.62	McMaster	1566 Carbon Steel
SlipRing 5 Channel (14)	Moog AC2690-5	1	NA	Moog	Gold and Others
Arm2 (15)	McMaster 8975K224	1	27.4	McMaster	Aluminum
97022A213 (16)	McMaster 97022A213	2	5.34*	McMaster	316 Stainless Steel
EncoderDisk (17)	US Digital HUBDISK-2 -10000-250-IE	2	74.16	US Digital	Others

Table C.5 continued from previous page

Designator	Component	Number	Total cost-currency	Source of materials	Material type
98126A447 (18)	McMaster 98126A447	2	6.82*	McMaster	18-8 Stainless Steel
97633A130 (19)	McMaster 97633A130	2	9.77*	McMaster	1060-1090 Spring Steel
BearingPlate (20)	McMaster 89015K255	2	31.7	McMaster	Aluminum
Arm1Cover (21)	3D Printed	1	NA	NA	PLA
Arm3Shaft (23)	McMaster 1346K11	1	8.62	McMaster	1566 Carbon Steel
Arm3 (23)	McMaster 8975K224	1	27.4	McMaster	Aluminum
Arm2Cover (24)	3D Printed	1	NA	NA	PLA

C.10.2 Bill of Materials for Building Pendulum Cart

All the materials needed to build the above mentioned pendulum cart can be seen in Table. C.6. Some of the components do have a public price information, thus we refer the reader to contact the corresponding manufacture and ask for a quote. As for some bolts and nuts, they are sold in packs. Thus, the price of one pack is listed and we indicate this using a "*" symbol. Besides the materials mentioned in Table. C.6, we also recommend ordering 1074-1095 Spring Steel Ring Shim (McMaster 98055A117: 0.1mm thick, 10mm ID. McMaster 98055A121: 0.5mm Thick, 10mm ID) to serve as replacement of 3D printed shim (35) if the printer precision is not high.

Table C.6: Materials needed to build the pendulum cart in Fig. 6.7.

Designator	Component	Number	Total cost-currency	Source of materials	Material type
91251A077 (4)	McMaster 91251A077	6	8.74	McMaster	Black-Oxide Alloy Steel
Encoder (7)	US Digital EM2-2-10000-I	1	43.51	US Digital	Others
DriverClip (25)	3D Printed	1	NA	NA	PLA
CableDriver (26)	US Digital PC4-H10	3	51.09	US Digital	Others
96144A111 (27)	McMaster 96144A111	6	21.24	McMaster	Black-Oxide Alloy Steel
3329A310 (28)	McMaster 3329A310	1	28.1	McMaster	Chrome-Plated Brass and Others
CartPlate (29)	McMaster 9246K486	1	26.69	McMaster	Aluminum
LinearMotor (30)	HIWIN LMX1K -SA12-1-2000 -PGS1-V103+HS	1	NA	HIWIN	Aluminum and Others
61900 [6] (31)	61900 Hybrid Ceramic Bearing	2	81.9	Ortech	Ceramic, Steel, and Others
Spacer (32)	3D Printed	1	NA	NA	PLA
9140T273 (33)	McMaster 9140T273	1	39.23	McMaster	Aluminum
98055A122 (34)	McMaster 98055A122	1	10.69*	McMaster	1074-1095 Spring Steel
Shim (35)	3D Printed	1	NA	NA	PLA

[6] 61900 Hybrid Ceramic Bearings (10 X 22 X 6). Bearing Material: Chorme Steel 52100. Ball Material: Silicon Nitride. Cage/Retainer: PTFE. Enclosure: Open. Rating(ABEC): 7. Bearing Internal Clearance: Tight. Radial Play C2. Lube: Dry.

Table C.6 continued from previous page

Designator	Component	Number	Total cost-currency	Source of materials	Material type
EncoderDisk2 (36)	US Digital HUBDISK-2 -10000-394-IE	1	37.06	US Digital	Others
90326A105 (37)	McMaster 90326A105	1	6.35	McMaster	Steel
BlockBase (38)	3D Printed	1	NA	NA	NA
9125A471 (39)	McMaster 91251A471	2	8.35*	McMaster	Black-Oxide Alloy Steel
BrushBlock-8 (40)	Moog AC259-8	1	NA	Moog	Gold and Others
LimitSwitch Plate1 (41)	McMaster 89015K113	1	4.35*	McMaster	Aluminum
90751A110 (42)	McMaster 90751A110	4	8.86*	McMaster	18-8 Stainless Steel
LimitSwitch Plate2 (43)	McMaster 89015K113	1	4.35*	McMaster	Aluminum
6000N312 (44)	McMaster 6000N312	2	3.96	McMaster	Stainless Steel
LimitSwitch Base (45)	3D Printed	1	NA	NA	PLA
90592A085 (46)	McMaster 90592A085	4	1.44*	McMaster	Steel
91290A318 (47)	McMaster 91290A318	2	14.37*	McMaster	Steel
91290A121 (48)	McMaster 91290A121	2	14.33*	McMaster	Alloy Steel
LimitSwitch (49)	Omron EE-SX671 (PNP)	2	64.06	Omron	Ohters

C.10.3 Bill of Materials for System Frame

The materials needed to build the system frame can be summarized in Table. C.7. It is worth noting that component $56, 57, 62$ and 63 already comes with fastener, so there's no need to purchase additional fasteners. Moreover, the component $50, 55, 58$ and 61 can be directly purchased with the desired length by ordering McMaster 5537T907. This can ease the manufacturing process but will increase the overall price.

Table C.7: Materials needed to build the pendulum cart in Fig. C.7.

Designator	**Component**	**Number**	**Total cost-currency**	**Source of materials**	**Material type**
Extrusion1-4 (50)	McMaster 5537T103-4	8	261.52	McMaster	6560 Aluminum
X-Connector (51)	McMaster 89015K255	4	63.4	McMaster	6061 Aluminum
AlignTool1 (52)	3D Printed	8	NA	NA	PLA
AlignTool2 (53)	3D Printed	8	NA	NA	PLA
Drop-In Nut (54)	McMaster 5537T271	16	34.56	McMaster	Zinc-Plated Steel
Extrusion2-4 (55)	McMaster 5537T103-4	4	130.76	McMaster	6560 Aluminum
47065T998 (56)	McMaster 47065T998	10	166.1	McMaster	Anodized Aluminum
47065T984 (57)	McMaster 47065T948	2	32.48	McMaster	Anodized Aluminum
Extrusion3 (58)	McMaster 5537T103-2	2	35.84	McMaster	6560 Aluminum
Extrusion4 (59)	Leftover material when producing (58)	2	NA	McMaster	6560 Aluminum
5537T323 (60)	McMaster 5537T323	20	101	McMaster	Zinc

Table C.7 continued from previous page

Designator	Component	Number	Total cost-currency	Source of materials	Material type
Extrusion5 (61)	McMaster 5537T103-2	2	35.84	McMaster	6560 Aluminum
Wheel (62)	McMaster 4706T83	4	83.52	McMaster	Zinc-Plated Steel and Rubber
5537T37 (63)	McMaster 5537T37	4	105.76	McMaster	Anodized Aluminum
Extrusion6 (64)	McMaster 5537T103-6	2	92.54	McMaster	6560 Aluminum

C.10.4 Bill of Materials for Real-Time System

This section summarizes the materials needed to build the real-time system shown in Fig.6.10. All materials needed is shown in Table. C.8. Since monitor (69) does not affect the entire system's performance, we will not include the detailed information on which monitor we used. As long as the monitor the readers choose to use has a display port, it can connect with the target system. The host computer we used is quite old, so we decided not to put its price. The readers can use any computer they like as a host computer if it supports running Matlab and has an Ethernet port. As for the price of Speedgoat product, please send a quote to Speedgoat for more details.

Table C.8: Materials needed to build the real-time system.

Designator	Component	Number	Total cost-currency	Source of materials	Material type
Terminal Block (65)	Speedgoat Terminal Board 2×17 Pin	2	NA	Speedgoat	Others

Table C.8 continued from previous page

Designator	Component	Number	Total cost-currency	Source of materials	Material type
Cable (66)	M12M to M12F 17 Pin 3 feet Cable	4	NA	Speedgoat	Others
IO-191 $(67-1)$	IO-191 -Baseline	1	NA	Speedgoat	Others
IO-392 $(67-2)$	IO-392 -Baseline	1	NA	Speedgoat	Others
TargetMachine $(67-3)$	Baseline Education Real-Time Target Machine	1	NA	Speedgoat	Others
HostComputer (68)	NA	1	NA	NA	Others
Monitor (69)	NA	2	NA	NA	Others
5537T698 (70)	McMaster 5537T698	2	2.14	McMaster	Silver Zinc-Plated Steel
8961K102 (71)	McMaster 8961K102	1	3.9	McMaster	Zinc-Plated Steel
IO-392 DAQ4RS422 (72)	IO-392 Configuration File	1	NA	Speedgoat	NA

C.10.5 Bill of Materials for the Electronic Parts

This section summarize necessary components needed in the electrical system of the multi-link pendulum on the cart system. All the components needed in the electrical parts are shown in Table. C.9. Some of the components needed such as cables, DIN rail, DIN rail terminal blocks , wires, etc, are not shown in Fig. 6.12 and is included in Table. C.9.

Table C.9: Materials needed to build the electrical part of the system.

Designator	Component	Number	Total cost-currency	Source of materials	Material type
EmergencyStop Module (75)	Powertec 71354	1	19.99	Powertec	Others
Circuit Breaker (76)	Eaton FAZ-B3 -2-NA	1	48	Automation Direct	Others
Magnetic Contactor (77)	WEG CWC016 -00-22V18	1	23.5	Automation Direct	Others
Noise Filter (78)	Schaffner FN2090 -10-06	1	31.5	Automation Direct	Others
Line Reactor (78)	LR-11P0 -1PH	1	139	Automation Direct	Others
Motor Drive (80)	HIWIN D1 Drive	1	NA	HIWIN	Others
Edge Filter (81)	HIWIN MF-40-S	1	NA	HIWIN	Others
Regenerative Resistor (82)	Yohii 500W 30 Ω Resistor	2	2.14	Amazon	Others
DC Supply (83)	MEAN WELL MDR-60-24	1	24.84	MEAN WELL	Others
Enclosure (84)	Hammond CSKO242410	1	171	Automation Direct	Carbon Steel

Table C.9 continued from previous page

Designator	Component	Number	Total cost-currency	Source of materials	Material type
Toggle Switch (85)	20A 12VDC / 120VAC Toggle Switch	1	7.99	Amazon	Others
Momentary Switch (86)	LC LICTOP AC 660V Push Button Switch	1	6.99	Amazon	Others
EMI Cores (87)	HIWIN D1-EMC1	2	NA	HIWIN	Others
Enclosure Panel (88)	Hammond CSFC2424	1	21	Automation Direct	Carbon Steel
DIN Rail (89)	DN-R35S1-2	1	12.5	Automation Direct	Plated Steel
14AWG Cable (90)	Amazon 14 AWG Cable	1	NA	Amazon	Copper
12AWG Cable (91)	Amazon 12 AWG Cable	1	NA	Amazon	Copper
10AWG Cable (92)	Amazon 10 AWGCable	1	NA	Amazon	Copper
DIN Rail Terminal Block (93)	Amazon DIN Rail Block	10	NA	Amazon	Others
DIN Rail Ground Terminal Block (94)	Amazon DIN Rail Terminal Block	10	NA	Amazon	Others

Table C.9 continued from previous page

Designator	Component	Number	Total cost-currency	Source of materials	Material type
DC Connector (95)	Wago 721-103/ 026-000	1	4.03	Newark	Others
AC Connector (96)	Wago 721-204/ 026-000	1	6.91	Digikey	Others
MotorPower Connector (97)	Wago 721-104/ 026-000	1	5.26	Digikey	Others
EdgeFilter Input Connector (98)	Phoenix Contact 1718517	1	11.3	Mouser Electronics Phoenix Contact 1718517	Others
EdgeFilter Output Connector (99)	Phoenix Contact 1718504	1	11.3	Mouser Electronics Phoenix Contact 1718504	Others
Regen Connector (100)	Wago 723-603	1	3.85	Galco	Others
Communication Cable (101)	D1-DNT07A USB232 to RJ11 Adapter Cable	1	22	Amazon	Others
HookUp Wire (102)	24 AWG Hook Up Wire	1	NA	Amazon	Copper

Table C.9 continued from previous page

Designator	Component	Number	Total cost-currency	Source of materials	Material type
Shielding Wire (103)	Electriduct 1/2" Tinned Copper Metal Braid Sleeving Flexible EMI RFI Shielding Wire Mesh (0.32" Diameter) - 10 Feet	1	21.88	Amazon	Tinned Copper Metal
Cable Sleeve (104)	Keco 100ft – 1/2 inch PET Expandable Braided Cable Sleeve	1	16.99	Amazon	Others
CN2 Connector (105)	3M 10126-3000	1	11.62	Digikey	Others
CN2 Connector Cover (106)	3M 10326-52F0-008	1	10.62	Digikey	Others
Limit Switch Cable (107)	HIWIN D-Sub 9-Pin Female Limit Switch Extension Cable	1	NA	HIWIN	Others

Appendix D

APPENDIX FOR CHAPTER 7

D.1 Governing Equations of the PCR3BP

In Section 7.2 we illustrated the tube dynamics of index-1 saddles using numerical simulations from the planar circular restricted three body problem (PCR3BP). We assume there are two massive bodies with masses μ_1 and μ_2 which form their own two body problem, while a (relatively) weightless body such as a satellite or shuttle moving in the gravitational field generated by these objects. Here we present the relevant dynamical system for the third weightless body restricted to the orbital plane of the two massive bodies which constitute the Hamiltonian PCR3BP system. We define the states of the system as x, y, v_x and v_y, representing the position of the third body in the plane and its velocity in each independent direction.

The potential energy of the system is given by

$$V_{3BP} = -\frac{1}{2}\left(\mu_1 r_1^2 + \mu_2 r_2^2\right) - \frac{\mu_1}{r_1} - \frac{\mu_2}{r_2}, \tag{D.1}$$

where

$$\begin{aligned} r_1^2 &= (x+\mu_2)^2 + y^2, \\ r_2^2 &= (x-\mu_1)^2 + y^2, \\ \mu &= \frac{m_2}{m_1+m_2}, \\ \mu_1 &= 1-\mu, \\ \mu_2 &= \mu. \end{aligned} \tag{D.2}$$

The kinetic energy is given by

$$T_{3BP} = \frac{1}{2}(v_x^2 + v_y^2), \tag{D.3}$$

so that the total energy of the PCR3BP is

$$\mathcal{H}_{3BP} = T_{3BP} + V_{3BP}. \tag{D.4}$$

Using the above Hamiltonian, we can arrive at the corresponding first-order ODE describing the equations of motion of the PCR3BP. The result is given by

$$\begin{aligned}
\dot{x} &= v_x, \\
\dot{y} &= v_y, \\
\dot{v}_x &= 2v_y + x - \frac{(x-(1-\mu))\mu_2}{r_1^3} - \frac{(1-\mu)(x+\mu)}{r_2^3}, \\
\dot{v}_y &= -2v_x + y - \frac{y\mu_2}{r_1^3} - \frac{\mu_1 y}{r_2^3},
\end{aligned} \tag{D.5}$$

Here μ is the mass ratio of the two massive bodies. For the examples provided in Section 7.2 we have followed [437] used taken $\mu = 9.537 \times 10^{-4}$, which is the mass ratio of Jupiter to the Sun in our solar system.

The L_1 and L_2 index-1 saddle Lagrangian points lie on the line $y = 0$. Such points are equilibria of (D.5) with $y = 0$ and, per [439, Section 2.5], are given by $x = (1-\mu) \pm \gamma$, where γ are solutions of the quintic polynomials

$$\gamma^5 \mp (3-\mu)\gamma^4 + (3-2\mu)\gamma^3 - \mu\gamma^2 \pm 2\mu\gamma - \mu = 0. \tag{D.6}$$

Precisely, the L_1 Lagrange point comes from a solution γ with the upper signs in the above polynomial, while the L_2 Lagrange point comes from using lower signs. With the value $\mu = 9.537 \times 10^{-4}$, the corresponding γ values in (D.6) are $\gamma_1 = 0.06667655835869524$ and $\gamma_2 = 0.06978018274821962$, thus leading to the x positions of the L_1 and L_2 Lagrangian

points

$$\begin{aligned} x_{L_1} &= (1-\mu) - \gamma_1 = 0.9323697416413048, \\ x_{L_2} &= (1-\mu) + \gamma_2 = 1.0688264827482197. \end{aligned} \tag{D.7}$$

The coordinates $(x_{L_i}, 0, 0, 0)$, $i = 1, 2$, thus represent the location of the index-1 saddles of interest in the PCR3BP (D.5).

D.2 The Point-Mass Double Pendulum

Here we will provide the relevant equations of motion for the point-mass double pendulum and describe the analogous linear analysis to that of § 7.4.2. The goal is to show that the Down-Up and Up-Down states remain index-1 saddles for the simplified point-mass double pendulum. Furthermore, we comment that although it is not reported here, there is a nearly identical tube structure to what we have shown for the experimental double pendulum, meaning that our election to study the more physically realistic experimental double pendulum presents little difference to the more typically studied point-mass double pendulum.

To begin, let m_1 denote the point mass of the first (base) arm and m_2 be the point mass of the second arm. We take the length of the first and second arms to be ℓ_1 and ℓ_2, respectively. The angle θ_1 represents the angular displacement of the first arm from the downward vertical position, while θ_2 is the angular displacement of the second arm. We again refer the reader to the left panel of Figure C.26 for visual reference.

The kinetic energy of the double pendulum is given by

$$T = \frac{1}{2}(m_1 + m_2)\ell_1^2\omega_1^2 + \frac{1}{2}m_2\ell_2^2\omega_2^2 + m_2\ell_1\ell_2\omega_1\omega_2\cos(\theta_2 - \theta_1) \tag{D.8}$$

and the potential energy is given by

$$V = -(m_1 + m_2)\ell_1 g\cos(\theta_1) - m_2\ell_2 g\cos(\theta_2). \tag{D.9}$$

Hence, the total energy of the system is given by the Hamiltonian function

$$\mathcal{H}(\theta_1, \theta_2, \omega_1, \omega_2) = T + V \tag{D.10}$$

composed of the sum of the kinetic (D.8) and potential (D.9) energies. Following as in Section C.9.1 we arrive at the corresponding first-order conservative ODE describing the equations of motion for the point-mass double pendulum:

$$\begin{aligned}
\dot{\theta}_1 &= \omega_1 \\
\dot{\theta}_2 &= \omega_2 \\
\dot{\omega}_1 &= \frac{A_1 + A_2 + A_3}{\ell_1^2 \ell_2 (m_1 + m_2 - m_2 \cos^2(\theta_2 - \theta_1))} \\
\dot{\omega}_2 &= \frac{B_1 + B_2 + B_3}{\ell_1 \ell_2^2 m_2 (m_1 + m_2 - m_2 \cos^2(\theta_2 - \theta_1))}
\end{aligned} \quad . \tag{D.11}$$

Here we have

$$\begin{aligned}
A_1 &= m_2 \ell_1 \ell_2^2 \omega_2^2 \sin(\theta_2 - \theta_1), \\
A_2 &= \frac{1}{2} m_2 \ell_1^2 \ell_2 \omega_1^2 \sin(2\theta_2 - 2\theta_1), \\
A_3 &= \ell_1 \ell_2 g [m_2 \sin(\theta_2) \cos(\theta_2 - \theta_1) - (m_1 + m_2) \sin(\theta_1)], \\
B_1 &= -\frac{1}{2} \ell_1 \ell_2^2 m_2^2 \omega_2^2 \sin(2\theta_2 - 2\theta_1), \\
B_2 &= -\ell_1^2 \ell_2 m_2 (m_1 + m_2) \omega_1^2 \sin(\theta_2 - \theta_1), \\
B_3 &= -\ell_1 \ell_2 m_2 g (m_1 + m_2) [\sin(\theta_2) - \sin(\theta_1) \cos(\theta_2 - \theta_1)],
\end{aligned} \tag{D.12}$$

where we have neglected any friction or control terms that could be added to the system.

We now follow the linear analysis of § 7.4.2 to show that again the Down-Up and Up-Down steady-states are index-1 saddles of (D.11). Again, equilibria come in the form $(\theta_1, \theta_2, \omega_1, \omega_2) = (\pi k_1, \pi k_2, 0, 0)$ for every pair of integers $(k_1, k_2) \in \mathbb{Z}^2$ and all of these equilibria are symmetric with respect to the action of the reverser (7.22). As with the

experimental double pendulum, by periodicity of the $\theta_1, 2$ components we have four equilibria that merit investigation: Down-Down, Down-Up, Up-Down, and Up-Up, as described in (7.23).

Linearizing (D.11) about an equilibrium $(\theta_1, \theta_2, \omega_1, \omega_2) = (\pi k_1, \pi k_2, 0, 0)$ with $k_1, k_2 \in \mathbb{Z}$ results in the matrix

$$\begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \frac{-g(m_1+m_2)\cos(\pi k_1)}{\ell_1(m_1+m_2-m_2\cos^2(\pi k_2-\pi k_1))} & \frac{gm_2\cos(\pi k_2)\cos(\pi k_2-\pi k_1)}{\ell_1(m_1+m_2-m_2\cos^2(\pi k_2-\pi k_1))} & 0 & 0 \\ \frac{g(m_1+m_2)\cos(\pi k_1)\cos(\pi k_2-\pi k_1)}{\ell_2(m_1+m_2-m_2\cos^2(\pi k_2-\pi k_1))} & \frac{-g(m_1+m_2)\cos(\pi k_2)}{\ell_2(m_1+m_2-m_2\cos^2(\pi k_2-\pi k_1))} & 0 & 0 \end{bmatrix}. \tag{D.13}$$

The block structure of (D.13) gives the square of its eigenvalues are equal to those of the lower left 2×2 matrix

$$\begin{bmatrix} \frac{-g(m_1+m_2)\cos(\pi k_1)}{\ell_1(m_1+m_2-m_2\cos^2(\pi k_2-\pi k_1))} & \frac{gm_2\cos(\pi k_2)\cos(\pi k_2-\pi k_1)}{\ell_1(m_1+m_2-m_2\cos^2(\pi k_2-\pi k_1))} \\ \frac{g(m_1+m_2)\cos(\pi k_1)\cos(\pi k_2-\pi k_1)}{\ell_2(m_1+m_2-m_2\cos^2(\pi k_2-\pi k_1))} & \frac{-g(m_1+m_2)\cos(\pi k_2)}{\ell_2(m_1+m_2-m_2\cos^2(\pi k_2-\pi k_1))} \end{bmatrix}. \tag{D.14}$$

We therefore arrive at the following results.

Down-Down. The Down-Down state is obtained by setting $k_1 = k_2 = 0$. In this case (D.14) becomes

$$\begin{bmatrix} \frac{-g(m_1+m_2)}{\ell_1 m_1} & \frac{gm_2}{\ell_1 m_1} \\ \frac{g(m_1+m_2)}{\ell_2 m_1} & \frac{-g(m_1+m_2)}{\ell_2 m_1} \end{bmatrix}. \tag{D.15}$$

Since the trace of the above matrix is negative and the determinant is positive for all $m_1, m_2, \ell_1, \ell_2 > 0$, it follows that both eigenvalues are distinct and strictly negative. Therefore, the linearization (D.13) about the Down-Down state has two pairs of purely complex eigenvalues, making it a linear center.

Down-Up. The Down-Up state is obtained by setting $k_1 = 0$ and $k_2 = 1$. The matrix (D.14)

becomes

$$\begin{bmatrix} \frac{-g(m_1+m_2)}{\ell_1 m_1} & \frac{gm_2}{\ell_1 m_1} \\ \frac{-g(m_1+m_2)}{\ell_2 m_1} & \frac{g(m_1+m_2)}{\ell_2 m_1} \end{bmatrix} \tag{D.16}$$

which has a negative determinant for all relevant parameter values. Therefore, the above matrix has one positive and one negative real eigenvalue, thus making the Down-Up state an index-1 saddle since (D.13) has one positive eigenvalue, one negative eigenvalue, a pair purely imaginary eigenvalues when $k_1 = 0$ and $k_2 = 1$.

Up-Down. We obtain the Up-Down state by setting $k_1 = 1$ and $k_2 = 0$. The matrix (D.14) is then given by

$$\begin{bmatrix} \frac{g(m_1+m_2)}{\ell_1 m_1} & \frac{-gm_2}{\ell_1 m_1} \\ \frac{g(m_1+m_2)}{\ell_2 m_1} & \frac{-g(m_1+m_2)}{\ell_2 m_1} \end{bmatrix} \tag{D.17}$$

and following as in the Down-Up state, we find that the Up-Down state is also an index-1 saddle.

Up-Up. The Up-Up state has $k_1 = k_2 = 1$, resulting in (D.14) taking the form

$$\begin{bmatrix} \frac{g(m_1+m_2)}{\ell_1 m_1} & \frac{-gm_2}{\ell_1 m_1} \\ \frac{-g(m_1+m_2)}{\ell_2 m_1} & \frac{g(m_1+m_2)}{\ell_2 m_1} \end{bmatrix}. \tag{D.18}$$

The above matrix is the result of negating all of the entries of the Down-Down analysis by -1. It follows that the linearization (D.13) about the Up-Up state has two positive eigenvalues and two negative eigenvalues. From the nomenclature above, the Up-Up state is an *index-2 saddle*.

From above we can see that the Down-Up and Up-Down states are index-1 saddles, similar to the experimental pendulum.