A Framework for Linear Prediction of Nonlinear Dynamical Systems Using Koopman Theory

Ashwin Srinivas Badrinath

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

Master of Science in Electrical Engineering

University of Washington

2020

Committee:

Sreeram Kannan,

Jose Nathan Kutz

Program Authorized to Offer Degree:
Electrical Engineering

University of Washington

Abstract

A Framework for Linear Prediction of Nonlinear Dynamical Systems Using Koopman Theory

Ashwin Srinivas Badrinath

Chair of the Supervisory Committee:
Professor Sreeram Kannan
Electrical Engineering

Despite many advances being made in classical techniques for handling dynamical systems, the class of nonlinear dynamical systems is yet to be treated under a "one size fits all" scheme as is the case with linear dynamical systems. But given that all linear systems lend themselves to easy representation, analysis and control, one could leverage existing theory that allows us to examine nonlinear dynamical systems under the same lens.

Koopman theory comes as an answer to this felt need of simplifying how we deal with nonlinear dynamical systems and influence their behaviour. By riding on the rising tides of big data and massive compute that is prevalent in our times, data driven methods to approximate the Koopman operator can be used to develop a framework to cast a nonlinear dynamical system into a linear dynamical system in a higher dimensional state space. This higher dimensional state space can be operated in and the resulting actions and trajectories that the system assumes in this higher state space can then be translated to the original manifold that the system lives in naturally.

This thesis proposes an end-to-end framework that constructs a linear approximation to a nonlinear dynamical system by lifting the original state space to a higher dimensional

state space where it is linear. The preprocessing stage that one must go through, the choice of lifting function that results in the higher dimensional state space, building the linear model in this higher dimensional state space and subsequently forecasting with an initial condition and some control inputs, if applicable, are all discussed.

Two methods were tried on three classes of systems which are unforced nonlinear dynamical systems, forced affine nonlinear dynamical systems and forced nonaffine nonlinear dynamical systems. One method leverage deep learning to choose the lifting functions to attain linear advancement in the resulting higher dimensional state space and the other makes use of sparse regression techniques to identify analytical expressions for possible candidate lifting functions, so in this case we are aware of what the lifting functions are exactly.

The trajectories that were obtained from the linear model derived in the higher dimensional state space were very close to the original trajectories obtained by advancing the nonlinear system with a numerical solver. This shows that this single framework is very reliable in representing and analyzing nonlinear dynamical systems.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

GLOSSARY

BACK-PROPAGATION (BACK-PROP): A method used for computing the gradient descent required for the training of neural networks. Based upon the chain rule, back-prop exploits the compositional nature of Neural Networks in order to frame an optimization problem for updating the weights of the network. It is commonly used to train deep neural networks.

DATA MATRIX: A matrix where each column vector is a snapshot of the state of a system at a particular instance in time. These snapshots may be sequential in time or they may come from an ensemble of initial conditions.

DEEP LEARNING: A class of machine learning algorithms that uses several hidden layers in Neural Networks with architectures ranging from Feedforward Neural Networks, Recurrent Neural Networks, Convolutional Neural Networks etc. Deep learning can leverage supervised (e.g., classification) and/or unsupervised (e.g., pattern analysis) algorithms, learning multiple levels of representations that correspond to different levels of abstraction. The levels form a hierarchy of concepts.

DICTIONARY LEARNING: A paradigm of representation learning wherein the feature vectors required for further stages of the machine learning problem are learned from the original data so as to transform the original data into this new learned feature space to obtain the required data driven objective.

DYNAMIC MODE DECOMPOSITION: The leading eigendecomposition of a best-fit linear operator $A = X'X^{\dagger}$ that propagates the data matrix X into a future data matrix X'. The eigenvectors of A are DMD modes and the corresponding eigenvalues determine the time dynamics of these modes.

DYNAMICAL SYSTEMS: Dynamical Systems is an overarching term to denote any system that changes with respect to some independent variable, usually time and/or space.

EXTENDED DYNAMIC MODE DECOMPOSITION: This is a variant of Dynamic Mode Decomposition where the data matrices are the original state measurements and some nonlinear functional on the original state space. Furthermore, the process makes use

of the right numerical method to account for the high dimensionality incurred in augmenting the original measurements with those of the functionals.

FORWARD-PROPAGATION (FORWARD-PROP): This refers to the process of transforming a data vector into a new vector by passing it through a trained neural network. This is usually done after the training process to assess the performance of the neural network.

HANKEL MATRIX: A matrix that has a special structure in that the following rows are time delayed or shifted versions of the the ones preceding them.

HILBERT SPACE: A generalized vector space with an inner product. When referred to in this text, a Hilbert space typically refers to an infinite-dimensional function space. These spaces are also complete metric spaces, providing a sufficient mathematical framework to enable calculus on functions.

KOOPMAN EIGENFUNCTION: An eigenfunction of the Koopman operator. These eigenfunctions correspond to measurements on the state space of a dynamical system that form intrinsic co-ordinates. In other words, these intrinsic measurements will evolve linearly in time despite the underlying system being nonlinear.

KOOPMAN EIGENMODE: These are the basis vectors along which the Koopman Operator advances the Koopman observables after scaling by the Koopman eigenfunctions and the Koopman eigenvalues.

KOOPMAN EIGENVALUE: These are the eigenvalues of the Koopman Operator.

KOOPMAN OBSERVABLE: A functional on the state space of a nonlinear dynamical system that produces a measurement that is part of the observable vector that is advanced linearly by the Koopman Operator.

KOOPMAN OPERATOR: An infinite dimensional linear operator that propagates measurement functions from an infinite dimensional Hilbert space through a dynamical system.

LINEAR DYNAMICAL SYSTEMS: Dynamical Systems that obey the principles of superposition and homogeneity.

NONLINEAR DYNAMICAL SYSTEMS: Dynamical Systems that do not obey the principles of superposition and homogeneity.

SNAPSHOT: A single high-dimensional measurement of a system at a particular time. A number of snapshots collected at a sequence of times may be arranged as column vectors in a data matrix.

SPARSE IDENTIFICATION OF NONLINEAR DYNAMICS (SINDY): A nonlinear system identification framework used to simultaneously identify the nonlinear structure and parameters of a dynamical system from data. Various sparse optimization techniques may be used to determine SINDy models.

STATE SPACE: The set of all possible system states. Often the state-space is a vector space, such as $\mathbf{R}^n$, although it may also be a smooth manifold $\mathbf{M}$.

STOCHASTIC GRADIENT DESCENT: Also known as incremental gradient descent, it allows one to approximate the gradient with a single data point instead of all available data. At each step of the gradient descent, a randomly chosen data point is used to compute the gradient direction.

SYSTEM IDENTIFICATION: The procedure by which a model representing the system's dynamics is identified by recording it's response to certain inputs or it's natural evolution under given initial conditions.

TIME DELAY COORDINATES: An augmented set of coordinates constructed by considering a measurement at the current time along with a number of times in the past at fixed intervals from the current time. Time delay coordinates are often useful in reconstructing attractor dynamics for systems that do not have enough measurements, as in the Takens embedding theorem.

ACKNOWLEDGMENTS

The author wishes to express sincere appreciation to the departments of Electrical Engineering and Applied Mathematics at the University of Washington for their flexibility, generosity and support towards the author's academic endeavors.

The author is also very grateful for the guidance and encouragement provided by the faculty members on the reading committee, namely, Prof. Sreeram Kannan and Prof. Nathan Kutz.

DEDICATION

to all those who have persevered and continue to persevere towards taming Nonlinear Dynamical Systems despite the chaos, stochasticity, high dimensionality and other tribulations.

Chapter 1

MOTIVATION

All linear systems assume the form $\dot{x} = Ax + Bu$. And just like that, the physics of the underlying system is abstracted away and one can apply general methods of analysis and control with just the A and B matrices. Hence, linear systems, when they manifest, are a treat to the scientist and/or engineer.

Unfortunately, the same is not the case with nonlinear systems that are generically of the form $\dot{x} = f(x, u)$. In this case, for the proposes of this thesis, they can further be divided as:

- Unforced Nonlinear Dynamical Systems : These are nonlinear dynamical systems without the control or forcing term, u at are generally represented as $\dot{x} = f(x)$.

- Forced Nonlinear Dynamical Systems: These are nonlinear dynamical systems with external control inputs being applied via some form of actuation. We could also say that these are "forced" by the forcing and/or control term u. When the forcing enters linearly, we say that the system is in control affine form and otherwise, we say that it is in control nonaffine form. These are generally denoted as shown below:

  - Control Affine Form: $\dot{x} = f(x) + Bu$
  - Control Non-affine Form: $\dot{x} = f(x, u)$

Our predecessors, have always been using classical techniques that relied heavily on analytical expressions and finding non-unique and situation specific solutions to identify and control these nonlinear dynamical systems and despite their prowess and efforts, we still have not

effectively, "solved" the problem of nonlinear control to be able to use a one size fits all solution as has long been developed for linear dynamical systems.

In the modern era, however, notions of Big Data and massive compute power are abound and is something our predecessors did not and could not leverage and this is where the opportunity to supplement and in some cases, supplant the previous techniques arises. With data driven methods and words like machine learning and AI finding their way across verticals, it comes as no surprise that the study of identifying and controlling nonlinear dynamical systems is no exception to this.

One useful way of universally approximating a nonlinear dynamical system is by using the Koopman Operator, after transforming the original state-space into a higher dimensional, possibly infinite dimensional, state space. This gives us a linear representation of the nonlinear dynamical system and we can then, apply the "one size fits all" solution and methods that come with linearity to these nonlinear dynamical systems in this higher dimensional state space. Whereupon availing of this, once any analysis, prediction or control is done in the higher dimensional state space, we need only project back down to the lower dimensional state space.

It is with the aforementioned idea that this thesis attempts to propose a framework to do exactly that: data driven identification of a linear representation of any nonlinear dynamical system, forced and unforced and subsequently verify the effectiveness of the system identification carried out with some linear forecasting with the attained linear model. As per this thesis, the ideal end-to-end framework will have three main elements in data preprocessing, building a linear model and subsequently forecasting. All of these are detailed below:

- Data Preprocessing : This is an important step in any data driven endeavor and mainly entails obtaining and cleaning up the data snapshots that will be used for in the stages

to come. In the case of dynamical systems, it is very often the case that we do not have access to all the states in the system in that some states cannot be measured and consequently, must be reconstructed from the states at hand. One way would be to use integrals and/or derivatives to obtain the missing states followed up with a stage of filtering to smooth out any noise incurred in the process. Another way would be to use time delay coordinates to construct a Hankel matrix and recover the states by doing dynamic mode decomposition on those. The data snapshots could be sequential or could stem from an ensemble of initial conditions. Ideally, the snapshots collected will be exhaustive enough in terms of trajectories passing through the subspace of the state space in which we wish to operate and analyze the system.

- Building the Linear Model : Once the data has been preprocessed and the data matrices are obtained, the linear approximation of the the nonlinear system is obtained using different methods. In this thesis, we explore how we can use extended Dynamic Mode Decomposition (eDMD) with a lifted state vector to build this linear model in these lifted co-ordinates so that it evolves linearly.

- Forecasting : Once the linear model is obtained, one need only supply an initial condition to the model with control inputs if any and use the system matrices to propagate the initial condition into a trajectory in time and possibly compare with the ground truth which either comes from data recordings of the system given that initial condition or using an ODE solver if the model is known.

Chapter 2

THEORY AND LITERATURE SURVEY

2.1 Theoretical Background

Dynamic Mode Decomposition

DMD can be thought of as an ideal combination of spatial dimensionality-reduction techniques, such as the proper orthogonal decomposition (POD) [**?**], with Fourier transforms in time [16]. Thus, correlated spatial modes are also now associated with a given temporal frequency, possibly with a growth or decay rate. The method relies simply on collecting snapshots of data x_k from a dynamical system at a number of times t_k , where $k = 1, 2, 3, ..., m$. DMD is algorithmically a regression of data onto locally linear dynamics $x_{k+1} = Ax_k$, where A is chosen to minimize $||x_{k+1} - Ax_k||_2$ over the $k = 1, 2, 3, ..., m-1$ snapshots. The advantages of this method are that it is very simple to execute and it makes almost no assumptions about the underlying system. The cost of the algorithm is a singular value decomposition (SVD) of the snapshot matrix constructed from the data x_k .

In practice, when the state dimension n is large, the matrix A may be intractable to analyze directly. Instead, DMD circumvents decomposition of A by considering a rank-reduced representation in terms of a Proper Orthogonal Decomposition (POD)-projected matrix $\tilde{A}$. The DMD method is explained as follows:

1. First, take the singular value decomposition (SVD) of X,i.e, $X \approx U\Sigma V$ and perform a low-rank truncation if necessary. That is, we take only the first r singular values and modes.

2. The matrix A may be obtained by using the pseudoinverse of X obtained via the SVD:

$$A = X'V\Sigma^{-1}U^*$$

In practice, it is more computationally efficient to compute $\tilde{A}$, the $r \times r$ projection of the full matrix A onto the POD modes:

$$\tilde{A} = U^*AU = U^*X'V\Sigma^{-1}$$

3. Compute the eigendecomposition of $\tilde{A}$:

$$\tilde{A}W = W\Lambda$$

4. We may reconstruct the eignedecomposition of A from W and Λ. In particular, the eigenvalues of A are given by Λ and the eigenvalues of A (DMD modes) are given by the columns of Φ:

$$\Phi = X'V\Sigma^{-1}W$$

5. We can now approximate the solution for all future times by using

$$x(t) \approx \sum_{k=1}^{r} \phi_k \exp(\omega_k t) b_k = \Phi \exp(\Omega t) b$$

where b is the vector of coefficients that tells us how much of each mode is present in the signal at the time and can be obtained from the initial conditions, i.e, at $t = 0$ from

$$b = \Phi^{\dagger} x_1$$

The Koopman Operator

The theory that enables the framework alluded to in this thesis dates back to 1931 with the release of the seminal paper on the Koopman operator for Hamiltonian Systems by Bernard Koopman [13].

Sparse Identification of Nonlinear Dynamics (SINDy)

While doing SINDy, we seek a model of the form $f(x) = \Theta(x)\xi$. Where $\Theta(x)$ is a library of functions of the state and can be given by, say,

$$\Theta(x) = \begin{bmatrix} 1 & X & X^2 & \dots & X^n & \sin(X) & \dots \end{bmatrix}$$

So, then, $\xi = \Theta(x)^{-1}\dot{X}$ and this can be obtained by a linear solver to promote sparsity like LASSO or by using least squares and thresholding to eliminate weak co-efficients.

Koopman Reduced Order Nonlinear Identification and Control(KRONIC)

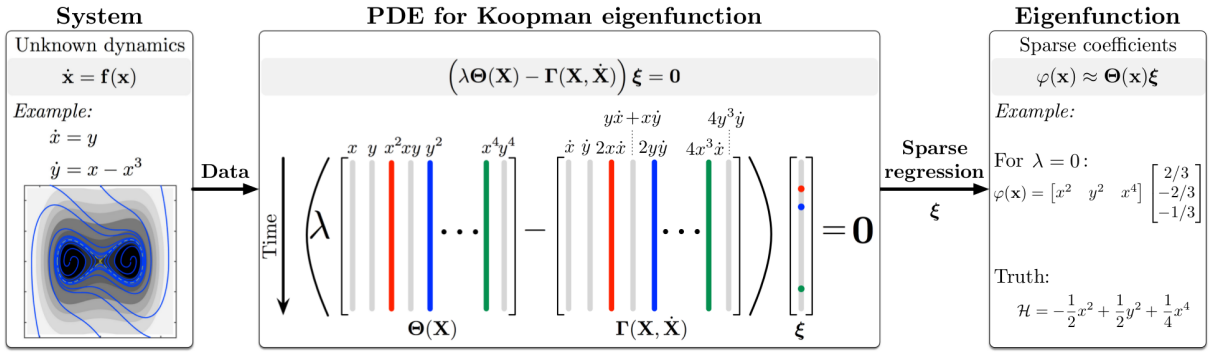

Figure 2.1: A SINDy like framework to discover eigenfunctions

Every eigenfucntion satisfies,

$$\frac{d}{dt}\varphi(x) = \lambda\varphi(x)$$

Applying the chain rule to the above equation we get

$$\frac{d}{dt}\varphi(x) = \nabla\varphi(x).\dot{x} = \nabla\varphi(x)f(x)$$

$$\nabla\varphi(x)f(x) = \lambda\varphi(x)$$

Now, we guess that these functions could reside in a library of functions given by

$$\Theta(X) = [\theta_1(X^T), \theta_2(X^T), \dots \theta_p(X^T)]$$

and the gradient is given by

$$\Gamma(X, \dot{X}) = [\theta_1(X^T)\dot{X}, \theta_2(X^T)\dot{X}, \ldots \theta_p(X^T), \dot{X}]$$

So, the equation that needs solving is now

$$(\lambda\Theta - \Gamma(X, \dot{X}))\xi = 0$$

2.2 Literature Survey

2.2.1 Linear Models for Nonlinear Dynamical Systems

Ever since the first paper on Dynamic Mode Decomposition (DMD)[21], there has been an increasing interest in using the central idea that it represents to identify and control nonlinear dynamical systems using a data driven approach. The DMD algorithm has also been found to yield an approximation of the Koopman Operator[3][19] which is particularly useful for the purposes of this thesis and in general as we can now use the effective and readily available tools that are used to analyze linear systems. But the vanilla DMD itself has issues pertaining to inaccuracy in approximating the Koopman operator and in only being limited to the state measurements themselves. Further ideas based on using nonlinear measurements [17][16] along with efficient numerical computation methods[16][23] have shown to yield much better approximations of the Koopman operator.

Quite a few approaches have posited some candidate lifting functions to serve as nonlinear measurements that can augment the original state space [23] but the lifting functions themselves were chosen on an empirical basis and leaves behind a vacuum in the area of choosing the best lifting functions analytically based on the data or system structure and providing theoretical backing for the same. To leverage the power of neural networks as function approximations as an answer to this has been explored[17] and has shown to yield promising results. But this takes away from knowing the analytical expressions of the lifting functions and leads to a less interpret-able model.

Hence, there is a need to develop techniques that lead to interpret-able and analytical expressions for the lifting functions. Furthermore, deep learning based techniques, while effective, must also be properly combined with linear prediction frameworks in an end-to-end fashion to produce models that are capable of forecasting from initial conditions in the original state space of the nonlinear dynamical system. One such linear prediction framework has been

explored[14] using radial basis functions as advocated by recent works [23] but the aforementioned problems of not having theory backed selection of lifting functions and performance inferiority compared to using neural networks still persists.

2.2.2 Neural Networks and Dynamical Systems

Deep learning has found it's way into the area of system identification of nonlinear systems directly by identifying the governing equations themselves[11] or something pertaining to them like the eigenfunctions of the accompanying Koopman operator or intrinsic coordinates associated with the nonlinear system [8]. And more importantly, in finding possible Koopman observables to serve as lifting functions to find the measurement vector that is advanced linearly by the Koopman operator[17].

Apart from the evident black-box optimization and approximation that neural networks provide, there are several upsides to using them as designing neural networks and tuning hyperparameters associated with a certain type of tasks is more principled than heuristic. Design examples where the concept of "residual neural networks" [11] which serve as better autoencoders to the data have been exploited in learning partial differential equations (PDEs)[11]. Autoencoder based networks have also been used to identify intrinsic coordinates and eigenfunctions [8]. Regular feedforward networks have shown to be good approximations for the Koopman observables in using eDMD to approximate the Koopman operator [17]. Feedforward architectures are the ones most relevant to this thesis and for general function approximation and leveraging their accuracy, they can be combined with linear prediction methods instead of using lifting functions that were chosen based on empirical results [14].

One must be mindful of the fact that neural networks are better suited to interpolation rather than extrapolation as illustrated in an example to identify high dimensional coordinates to linearize PDEs [11]. Hence, while training, a neural network for any of the aforementioned tasks, one must ensure that there are enough number of data points and that the set of data

points include trajectories that cover the subspace of the state space to be operated in as exhaustively as possible.

2.2.3 Sparse Regression Based Frameworks

Given snapshots of the data from the system in a format that is similar to that required by the DMD algorithm, one can obtain analytical expressions for the system given a library of functions. A framework to implement exactly this has been used for a number of systems with explicit and implicit formulations[5][18] and goes by the moniker, "Sparse Identification of Nonlinear Dynamics" (SINDy). The problem is solved using sparse regression methods to extract out the useful terms in a library of candidate functions. One must be mindful of how the terms are chosen and spurious entries eliminated as it is not as simple as examining the magnitude of the non-zero coefficients [15].

But this SINDy like framework can also be incorporated into other approaches where analytical expressions for certain objects of interest are required. Examples include, incorporating the framework into the cost function of a neural network trained to learn eigenfunctions or intrinsic coordinates of the system [8] and using the properties of eigenfunctions of the Koopman operator associated with a dynamical system to identify analytical expressions for them based on the eigenvalues[12].

Another object of interest would be the Koopman observables themselves, which are contained in the eigenfunctions and lend themselves to easy generation once a pair or more of observables are obtained [16]. KRONIC can be used to generate some eigenfunctions but these eigenfunctions being principal eigenfunctions is contingent upon the corresponding eigenvalues being the principal eigenvalues. The observables, however are simpler to approximate and fit in better to help construct the lifted state vector that will be linearly advanced by the Koopman operator. And it is this concept that is yet to be properly explored and holds promise, if properly done, to not only yield the best observables that correspond to

the system but also analytical expressions for these lifting functions that will lead to highly interpret-able physics models.

2.2.4 Data Preprocessing

More often than not, not all the states of the nonlinear dynamical system of interest are measurable. Furthermore, they could be noisy and unevenly spaced in time among other practicalities one encounters in the process of experimentation and data collection. For the purposes of denoising, commonly used filtering techniques like mean filters, median filters etc should do the trick. The problem of uneven time stamps in the context of DMD can be alleviated by using a variant of the DMD commonly referred to as optimized DMD[1].

But the question of unveiling missing coordinates and latent variables is more of a challenge and is commonly tackled by the use of time delay coordinates [22] to create Hankel matrices [2] that are then put through matrix decompositions such as the SVD, DMD etc. But in the process of creating time delay embeddings, one is faced with the question of choosing the time delay parameter and the embedding dimension. These serve as parameters that need to be tuned. General consensus on the embedding dimension is that it should be greater than twice the number of original states [7]. To choose the time delay parameters, on the other hand, is more of an open problem that has resulted in different methods being put forth [9][10].

Chapter 3

METHODOLOGY : THE OVERARCHING FRAMEWORK

3.1 Data Preprocessing

To perform time delay embeddings, a Hankel matrix [4], H, was constructed where each state was entered stacked row after row and then the time stepped version of the states is stacked the row below that and the stacking continues with more and more time stepped versions of the states.

$$H = \begin{bmatrix} x_0 & x_1 & x_2 & \dots & x_{n-m} \\ y_0 & y_1 & y_2 & \dots & y_{n-m} \\ x_1 & x_2 & x_3 & \dots & x_{n-m-1} \\ y_1 & y_2 & y_3 & \dots & y_{n-m-1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_m & x_{m+1} & x_{m+2} & \dots & x_n \\ y_m & y_{m+1} & y_{m+2} & \dots & y_n \end{bmatrix}$$

And H' is obtained in the same way X' is obtained from X. These are then our new data matrices and we pass them into the DMD function to obtain new dynamics in a higher state space and then pull out the state variables of interest.

Note that if the control input is also to be included in the time delay embedding process, then we would follow the same convention as followed in Section 3.2 where an augmented state vector $\chi_k = [x_k, u_k]^T$.

3.2 Building the Linear Model

Once the original state vector is obtained, in its full form an/or using time delay embeddings, it must be lifted to a higher dimensional space where it is linear. To achieve this we, are effectively trying to choose the right observables to augment the state vector such that the lifted dimensions are advanced linearly by the Koopman operator such that they evolve along these basis vectors that are invariant under the action of the Koopman operator. So, given the most generic representation of a discrete nonlinear dynamical system as below

$$x_{k+1} = f(x_k, u_k) \tag{3.1}$$

We seek to obtain the approximation given by

$$z_{k+1} = Az_k + Bu_k \tag{3.2}$$

$$\hat{x}_k = Cz \tag{3.3}$$

where the lifted dimension is given by

$$z_k = \begin{bmatrix} \psi_1(x_k) \\ \psi_2(x_k) \\ \vdots \\ \psi_N(x_k) \end{bmatrix} \tag{3.4}$$

where $\psi_i : \mathbf{R}^n \to \mathbf{R}$, $N >> n$. Now, given the initial condition x_0 and the sequence of control inputs, $\mathbf{u} = \{u_1, u_2, ..., u_N\}$, we seek to forecast future values of the state using the predictor of the form (3.2)[14].

To properly generalize to controlled systems, we must consider the augmented state vector given by $\chi_k = \begin{bmatrix} x_k \\ u_k \end{bmatrix}$. So, we now have

$$\chi_{k+1} = F(\chi_k) = \begin{bmatrix} f(x_k, \mathbf{u}(\mathbf{0})) \\ Su_k \end{bmatrix} \tag{3.5}$$

where S is the shift operator $Su_i = u_{i+1}$. The Koopman operator $\mathcal{K} : \mathcal{H} \to \mathcal{H}$ associated with the augmented state equation is defined by $(\mathcal{K}\phi)(\chi_k) = \phi(F(\chi_k))$ for each $\phi : \mathbf{R}^n \times l(U) \to \mathbf{R}$ belonging to some space of observables $\mathcal{H}$.

As per the eDMD algorithm the equation that is used to approximate $\mathcal{K}$ is given by

$$\sum_{j=1}^{K} \|\phi(\chi_{k+1}) - A\phi(\chi_k)\|_2^2 \tag{3.6}$$

where $\phi(\chi_k) = \Big[\phi_1(\chi_k), ..., \phi_N(\chi_k)\Big]^T$ In order to obtain a linear predictor and a computable objective function in (3.6), we impose that the fucntions ϕ_i are of the form

$$\phi_i(\chi, \mathbf{u}) = \psi_i(\chi) + \mathcal{L}_i(\mathbf{u}) \tag{3.7}$$

where $\psi_i : \mathbf{R}^n \to \mathbf{R}$ is in general nonlinear but $\mathcal{L}_i : l(U) \to \mathbf{R}$ is linear. Without loss of generality (by linearity and causality), we can assume that $N_\phi = N + m$ for some $N > 0$ and that the vector of lifting functions $\phi = \Big[\phi_1, ..., \phi_{N_\phi}\Big]$ is of the form

$$\phi(\chi, \mathbf{u}) = \begin{bmatrix} \psi(\chi) \\ \mathbf{u}(\mathbf{0}) \end{bmatrix} \tag{3.8}$$

where $\psi = \Big[\psi_1, ..., \psi_N\Big]^T$ and $\mathbf{u}(0) \in \mathbf{R}^m$ denotes the first component of the sequence $\mathbf{u}$. Since we are not interested in predicting future values of the control sequence, we can disregard the last m components of each term $\psi(\chi_{j+1}) - A\psi(\chi_j)$ in (3.6). Denoting $\bar{A}$ the first N rows of A and decomposing this matrix such that $\bar{A} = [A, B]$ with $A \in \mathbf{R}^{n \times N}, B \in \mathbf{R}^{N \times m}$ and using the notation $\chi_j = (x_j, u_j)$ in (3.6), leads to the minimization problem

$$\min_{A,B} \sum_{j=1}^{K} \|\psi(\chi_{j+1}) - A\psi(\chi_j) - B\mathbf{u}_j(0)\|_2^2 \tag{3.9}$$

Minimizing the above equation over A and B leads to the required predictor of the form (3.2) starting fromthe initial condition in (3.4). The matrix C is obtained simply as the best

projection of χ onto the span of ψ_i's in a least square sense,i.e, as the solution to

$$\min_{C} \sum_{j=1}^{K} \|\chi_j - C\psi(\chi_j)\|_2^2 \tag{3.10}$$

Now, one is left with finding the numerical algorithm that helps finding the matrices A, B and C.

We assume that a set of data

$$X = \begin{bmatrix} x_1, ..., x_k \end{bmatrix}, Y = \begin{bmatrix} y_1, ..., y_k \end{bmatrix}, U = \begin{bmatrix} u_1, ..., u_k \end{bmatrix} \tag{3.11}$$

satisfying the relation $y_i = f(x_i, u_i)$ is available. Note that we do not assume any temporal ordering of the data. In particular, the data is not required to come form one trajectory of the nonlinear system.

Given the data X, Y, U as shown above, the matrices $A \in \mathbf{R}^{N\times N}$ and $B \in \mathbf{R}^{N\times m}$ in (3.2) are obtained as the best linear one-step predictor in the lifted space in a least squares sense,i.e, they are obtained as the solution to the optimization problem

$$\min_{A,B} \|Y_{lift} - AX_{lift} - BU\|_F \tag{3.12}$$

where

$$X_{lift} = \begin{bmatrix} \psi(x_1), ..., \psi(x_k) \end{bmatrix}, Y_{lift} = \begin{bmatrix} \psi(y_1), ..., \psi(y_k) \end{bmatrix} \tag{3.13}$$

with

$$\psi(x) := \begin{bmatrix} \psi_1(x) \\ \vdots \\ \psi_N(x) \end{bmatrix} \tag{3.14}$$

being a given dictionary of nonlinear functions or candidate Koopman observables. The matrix $C \in \mathbf{R}^{n\times N}$ is obtained as the best linear least-squares estimate of $\mathbf{X}$ given $\mathbf{X_{lift}}$,i.e., the solution to

$$\min_{C} \|X - CX_{lift}\|_F \tag{3.15}$$

The analytical solution to (3.12) is

$$\begin{bmatrix} A, & B \end{bmatrix} = Y_{lift}[X_{lift}, U]^{\dagger} \tag{3.16}$$

where † denotes the Moore-Penrose pseudoinverse of a matrix. The analytical solution to (3.15) is

$$C = XX_{lift}^{\dagger} \tag{3.17}$$

There are however, some practicalities to be considered in computing the solution to (3.16). It is not beneficial to use least-squares to solve this. In particular, for larger datasets with $K >> N$, it is beneficial instead, to solve the normal equations associated with (3.12). The normal equations read

$$V = \mathcal{M}G \tag{3.18}$$

with variable $\mathcal{M} = [A, B]$ and data

$$G = \begin{bmatrix} X_{lift} \\ U \end{bmatrix} \begin{bmatrix} X_{lift} \\ U \end{bmatrix}^T, V = Y_{lift} \begin{bmatrix} X_{lift} \\ U \end{bmatrix}^T \tag{3.19}$$

Any solution to (3.18) is a solution to (3.12).

Given the general procedure of constructing a linear predictor from data, the most difficult part is to choose the right functions to construct the lifted state vector. This thesis examines two ways of choosing these observables, one leveraging deep learning to learn a dictionary as in [17] and the other is based on the method outlined in [12] to use a SINDy like framework and exploiting the structural properties of Koopman eigenfunctions.

3.2.1 Method 1 - Dictionary Learning Based Lifting Functions

In this approach, learning the lifted state vector corresponding to (3.14) is accomplished by using a feedforward neural network as depicted in Figure 3.1.

The inputs are fed into the neural network which will output the lifted state vector corresponding to the state vector that was input after one forward pass. So, in effect, the neural network is a function approximator represented by $\Psi(x, \theta)$ where θ represents the parameters to be learned, i.e, the weights and biases of the neural network based on a cost function.

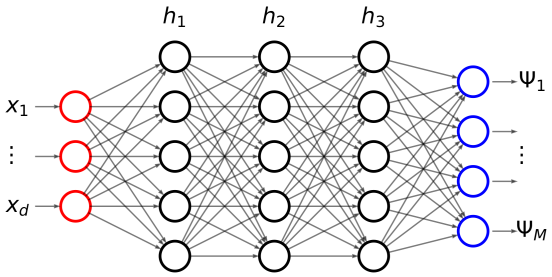

Figure 3.1: The feedforward neural network used for learning the dictionary of functions to create the lifted state vector

In this thesis, three hidden layers were used as is consistent from Figure 3.1 obtained from [17]. Hence, we can parameterize our function approximator as

$$\Psi(x) = W_{out} h_3 + b_{out}, \tag{3.20}$$

$$h_{k+1} = \tanh(W_k h_k + b_k), k = 0, 1, 2 \tag{3.21}$$

With Ψ parameterized, we can then define and solve the cost function

$$(K, \theta) = \text{argmin} J(\tilde{K}, \tilde{\theta}) \tag{3.22}$$

$$= \sum_{n=1}^{N} \|\Psi(y(n), \tilde{\theta}) - \tilde{K}\Psi(x(n), \tilde{\theta})\|^2 + \lambda \|\tilde{K}\|_F^2 \tag{3.23}$$

This is the cost function that will be used for stochastic gradient descent to train the parameters of the neural network,i.e, our function approximator. But if there exists a $\tilde{\theta}$ for which $\Psi(x, \theta) \equiv 0$, the right hand sie identically vanished and the minimum is trivially attained. Thus, to obtain meaningful approximations, we need further restrictions. A natural one is

to include $\Psi = \{\psi_1, ..., \psi_M\}$ some fixed (non-trainable) functions, such as the constant and the projection maps. The presence of the latter is important for reconstructing trajectories. This is because to find the Koopman modes we require the identity map $O(x) = x$, whose components are projection maps, to be in the linear span $U(\Psi) = \{\psi_1, \psi_2, ..., \psi_M\}$. The inclusion of these non-trainable dictionary functions then removes the possibility that $\Psi(x, \theta) \equiv 0$.

So, a practical algorithm to train the neural network would be:

Step 0: Randomly initialize K and θ.

Step 1: Fix θ, and optimize K. This can be done by solving the first term in (3.23),i.e, without the regularization. The analytical solution would be

$$K = (G(\tilde{\theta}) + \lambda I)^{\dagger} A(\tilde{\theta}) \tag{3.24}$$

where $G = \frac{1}{N}\sum_{n=1}^{N} \Psi(x(n), \tilde{\theta})^T \Psi(x(n), \tilde{\theta})$, $A = \frac{1}{N}\sum_{n=1}^{N} \Psi(x(n), \tilde{\theta})^T \Psi(y(n), \tilde{\theta})$ and I is an identity vector that matches the dimensions of G.

Step 2: Fix K, optimize θ. This is done by iteratively solving the full cost function in (3.23) using stochastic gradient descent and the expression for this is given by

$$\tilde{\theta} \leftarrow \tilde{\theta} - \delta \nabla_{\theta} J(\tilde{K}, \tilde{\theta}) \tag{3.25}$$

Step 3: Repeat Steps 1 and 2 until the cost function yields a low enough threshold or remains constant over several attempts. This would mean that if $J(K, \theta) < \epsilon$ or $J(K, \theta) = \epsilon$ for some small enough ϵ so that the forecasting is done reliably, the training process can be stopped.

Once this is done, one can generate the required lifted state vector as in (3.4) from the neural network and proceed as described in the main section 3.2.

3.2.2 Method 2 - KRONIC Based Lifting Functions

In this method, we attempt to derive analytical expressions for the entries in the lifted state vector. But KRONIC, in its first published form[12] has been used to identify analytic expres-

sions for eigenfunctions from data given some eigenvalues. That is, given those eigenvalues and the data readings, the eigenfunctions corresponding to those eigenvalues are extracted. But what we are interested in, in the framework, are the Koopman observables as those are the required functionals to preferably constitute our higher dimensional state space where the lifted vector is linearly advanced by the Koopman operator.

But the eigenfunctions have the interesting property that they are linear combinations of the Koopman observables [16]. This is evident from the following theorem:
Let φ_k be an eigenfunction of $\mathcal{K}$ with eigenvalue λ_k , and suppose $\varphi_k \in span\{g_j\}$, so that

$$\varphi_k(x) = w_1 g_1(x) + w_2 g_2(x) + \cdots + w_p g_p(x) = \mathbf{w}.\mathbf{g}$$

for some $\mathbf{w} = \begin{bmatrix} w_1, w_2, ..., w_p \end{bmatrix}^T \in \mathbf{C}^p$. If $w \in R(Y)$ where R is the range then $\mathbf{w}$ is a left eigenvector of A_Y with eigenvalue λ_k so that $\tilde{w}^* A_Y = \lambda_k \tilde{w}^*$.

We also make use of the property of eigenfunctions that states that given any pair of set of eigenfunctions and eigenvalues, more eigenpairs (eigenvalues and eigenfunctions) can be synthesized by multiplying the eigenfunctions together and adding the eigenvalues together. That is, if (λ_i, φ_i) and (λ_j, φ_j) are some given eigenpairs, then combinations like $(\lambda_i + \lambda_j, \varphi_i \varphi_j)$ are also eigenpairs[16][4]. So, what this means for the Koopman observables is that given a set of observables, more observables can be synthesized by multiplying them among themselves. Furthermore, in the analytic expressions for eigenfunctions, the individual terms are observables. So, we use KRONIC to identify not only the eigenfunctions but also the observables.

3.3 Forecasting

The process of forecasting is straightforward once the linear model is developed. Given the initial condition x_0 and the control inputs $[u_1, u_2, ..., u_k]$,if applicable, we can simply use our linear predictor to start time stepping from the initial condition adding in the control input

along the trajectory.

This can be double checked with the original system's trajectories with the same initial condition but forward propagated by a numerical method or if the dynamics are not known, a separate sample trajectory arising from an initial condition that was not trained on can be used for reference.

Chapter 4

EXPERIMENTS AND RESULTS

4.1 Method 1

4.1.1 Forced Nonlinear Dynamical System - Affine Case

The system considered for the affine case is a standard Vander Pol oscillator. The equations are given below in (4.1) and (4.2). In this example, both states are being observed and this precluded the necessity of any time delay embeddings.

$$\dot{x_1} = 2x_2 \tag{4.1}$$

$$\dot{x_2} = -0.8x_1 + 2x_2 - 10x_1^2x_2 + u \tag{4.2}$$

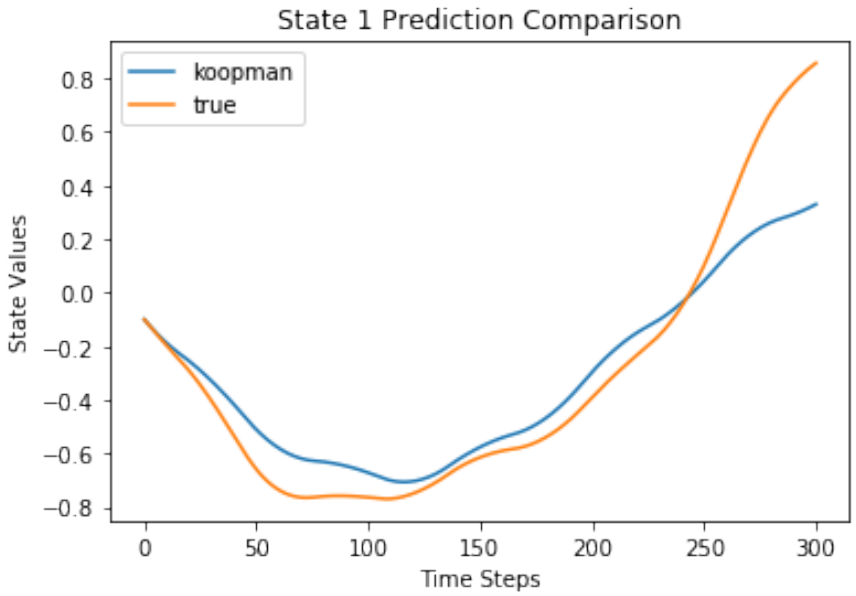

Figure 4.1: Results of state 1 of the Vander Pol Oscillator

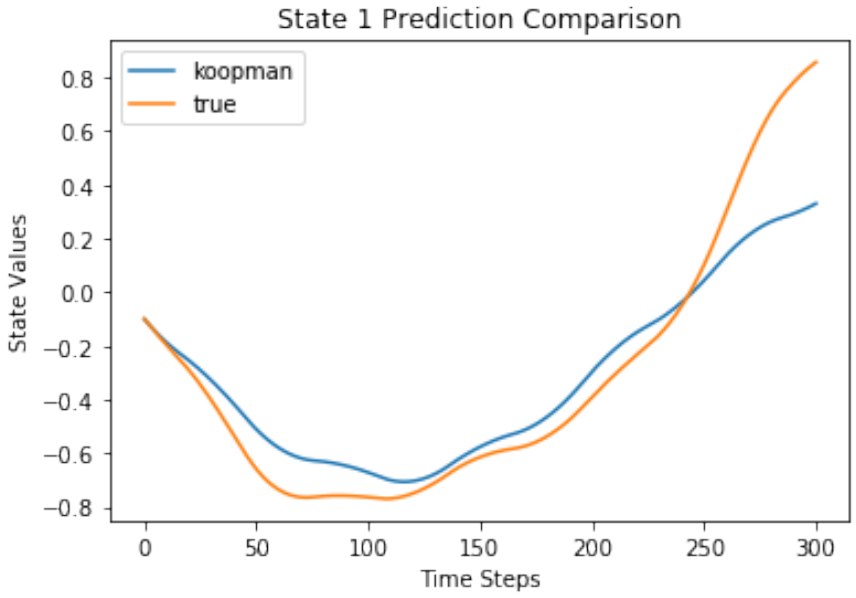

Figure 4.2: Results of state 2 of the Vander Pol Oscillator

Input	**Hidden Layer 1**	**Hidden Layer 2**	**Hidden Layer 3**	**Output**
2	52	52	52	26

Table 4.1: Neural Network Architecture for Lifting Vander Pol States (Dimension of Each Layer)

The neural network which lifted the states, did so from $\mathbf{R}^3$ to $\mathbf{R}^{26}$ and the dimensions of the hidden layers were designed to be twice that of the expected dimension of the lifted vector.

4.1.2 Forced Nonlinear Dynamical System - Nonaffine Case

The system considered for the affine case is a standard Vander Pol oscillator. The equations are given below in (4.3), (4.4) and (4.5). In this case, only state number 2,i.e, x_2 is observed. So, for this case, a time delay embedding of time shift value 1 was used. The Hankel matrix

had the form

$$H = \begin{bmatrix} y_k & y_{k-1} & \cdots & y_1 \\ y_{k-1} & y_{k-2} & \cdots & y_0 \\ u_{k-1} & u_{k-2} & \cdots & u_0 \end{bmatrix}$$

$$\dot{x_1} = -(R_a/L_a)x_1 - (k_m/L_a)x_2u + u_a/L_a \tag{4.3}$$

$$\dot{x_2} = -(B/J)x_2 + (k_m/J)x_1u - \tau_1/J \tag{4.4}$$

$$y = x_2 \tag{4.5}$$

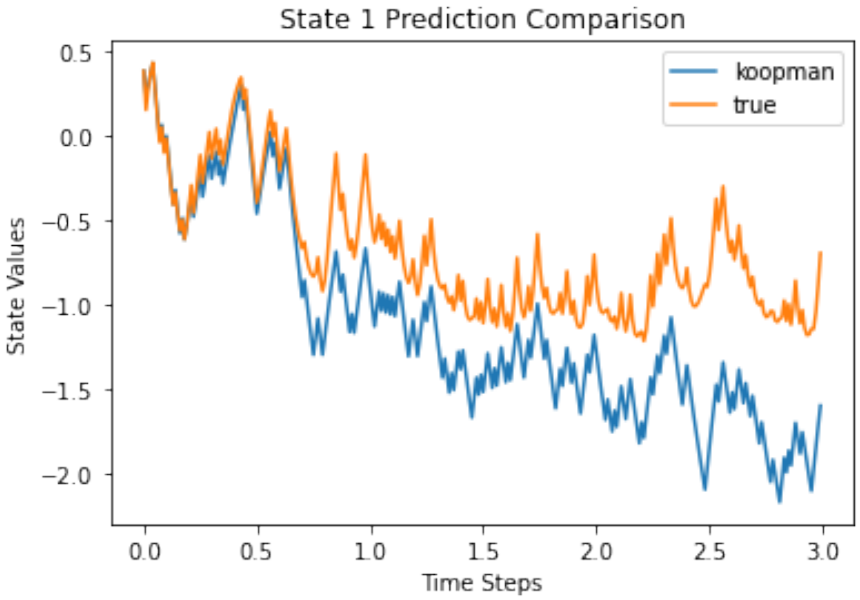

Figure 4.3: Results of observed state of the bilinear model of a DC motor with one sample test trajectory

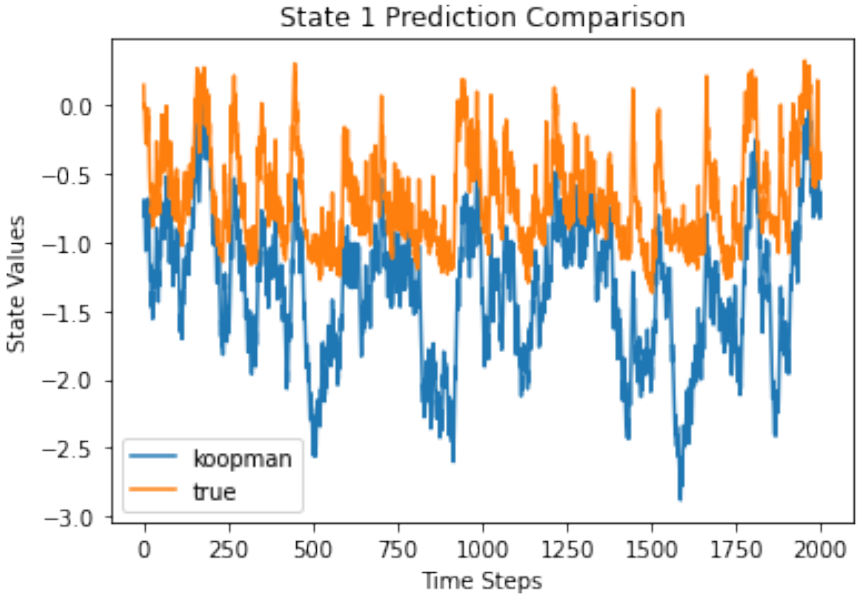

Figure 4.4: Results of observed state of the bilinear model of a DC motor with another sample test trajectory

Input	**Hidden Layer 1**	**Hidden Layer 2**	**Hidden Layer 3**	**Output**
3	58	58	58	29

Table 4.2: Neural Network Architecture for Lifting Motor States(Dimension of Each Layer)

The neural network which lifted the states, did so from $\mathbf{R}^3$ to $\mathbf{R}^{29}$ and the dimensions of the hidden layers were designed to be twice that of the expected dimension of the lifted vector.

4.2 Method 2

4.2.1 Unforced Nonlinear Dynamical System

Consider the System Given by the Equations

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \end{bmatrix} = \begin{bmatrix} \mu x_1 \\ \lambda(x_2 - x_1^2) \end{bmatrix} \tag{4.6}$$

Whose Principal Eigenfunctions are:

$$\phi_1(x) = x_1$$

$$\phi_2(x) = x_2 - \frac{\lambda}{\lambda - 2\mu} x_1^2$$

And the Rest of the Eigenfunctions are Obtained as:

$$\phi_i = x_1^{m_i^{(1)}} \left(x_2 - \frac{\lambda}{\lambda - 2\mu} x_1^2 \right)^{m_i^{(2)}}, i = 3, \ldots$$

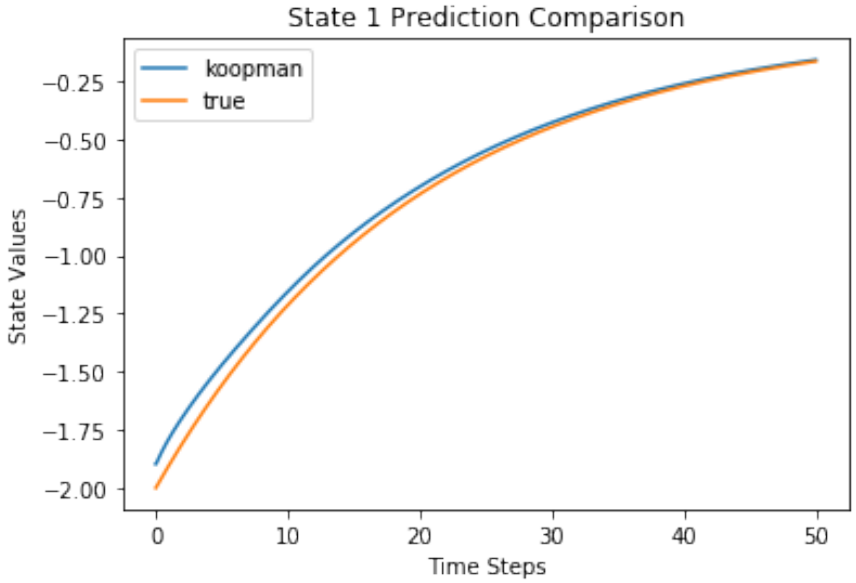

Figure 4.5: Results of state 1 of the Nonlinear System Under Consideration

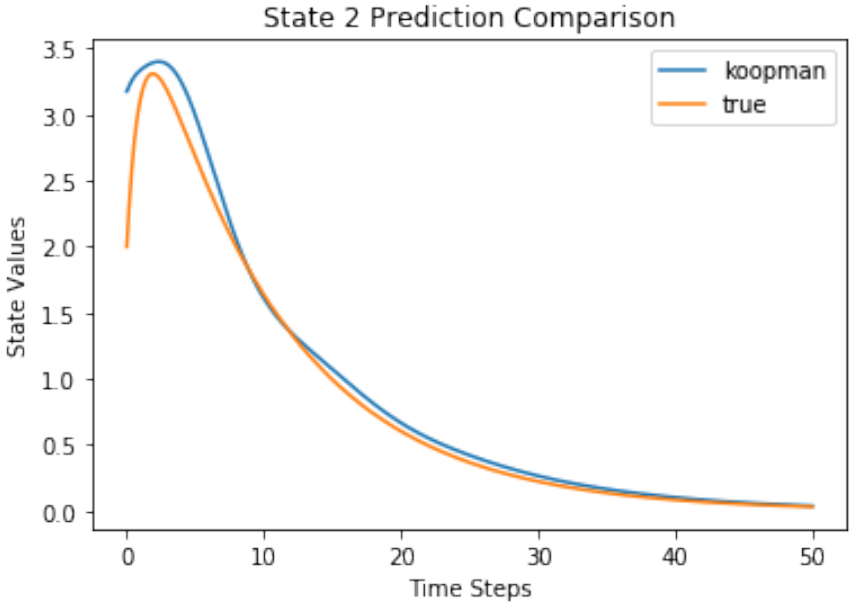

Figure 4.6: Results of state 2 of the Nonlinear System Under Consideration

An augmented state vector was made for each snapshot that included up to 20 Koopman observables generated from the principal eigenfunctions. The results are promising as shown in Figures 4.5 and 4.6. The test trajectory is traced by the linear predictor quite well.

For this example, the system is parameterized with $\lambda = -1$ and $\mu = -0.05$.

Chapter 5

CONCLUSIONS AND FUTURE WORK

5.1 Autoencoder Based Lifting of States Back and Forth

Given that eDMD-DL has shown so much promise in producing the right feature vectors that will go into the building the linear model, one can only conjecture that this approach can be used to build an auto-encoder architecture based neural network, similar to [8][11], that can help transform the state vector from it's original state space into a higher dimensional state space such that the higher dimensional state space is comprised of Koopman observables thereby lending itself to linear advancement under the action of the Koopman operator and then back down into the original state space. This could work better than the simple least squares problem being illustrated in Section 3.2.

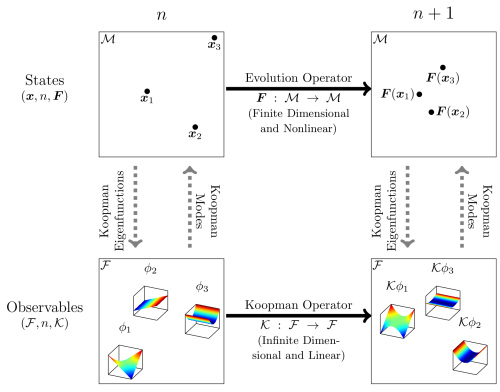

Figure 5.1: An Illustration of How the Proposed Framework Has Two sets of Manifolds to Operate In

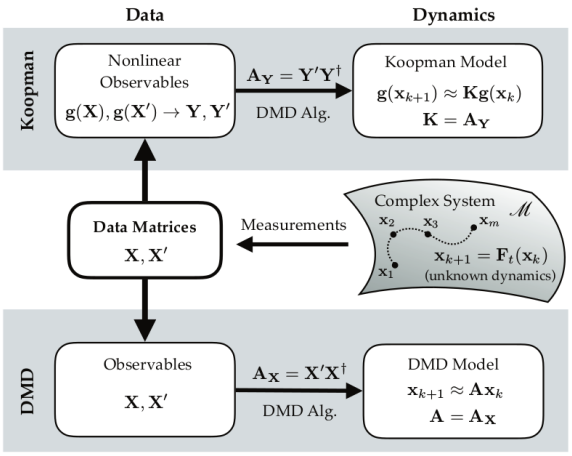

Figure 5.2: Approximating the Koopman Operator with Nonlinear Observables

Figures 5.1 and 5.2 show how flitting back and forth between the low dimensional nonlinear manifold and higher dimensional vector space would work in a highly illustrative and intuitive manner. Indeed, the route taken to go from to another must be chosen carefully so as to obtain optimal results and this is one step in that direction.

5.2 Incorporating Control Into the Framework

Given that the predictor being constructed in linear, one can now exploit the readily available and well studied methods for analysis and control of linear systems. Given the multivariable nature of resulting nonlinear system. Control techniques like the LQR, $\mathcal{H}_\infty$ and Model Predictive Control (MPC) must be tried in these higher dimensional coordinates to generate controllers which must then be transformed into the original state space.

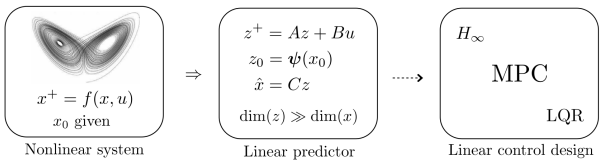

Figure 5.3: An Illustration of How Linear Control Techniques can be Incorported Into the Framework

5.3 Refining Method 2 to Cover All Relevant Cases

As seen in the KRONIC paper[12], the eigenfunction expressions and discovery process is dependent on the form the nonlinear system assumes. For nonlinear systems in control affine form, the eigenfunctions lend themselves to discovery in the same form but one must account for control just as how SINDy with control [6] accounts for the possibility of external forcing which is otherwise neglected in SINDy[5].

Similarly, the eigenfunctions discovered for non-affine nonlinear systems will also be non-affine in the sense that there will be dependency on the state and the control input just like in the system dynamics equations,i.e, $\varphi(x, u)$ will be the form of the eigenfunction.

Whether or not the true eigenfunctions of the unforced dynamics can be separated from snapshots with forcing also remains to be explored.

5.4 Finding Principal Eigenvalues Associated with a Given Koopman Operator

In Method 2 of the framework, it was shown that one need only obtain a finite set of eigenpairs to generate an infinite number of eigenpairs. This leads us to believe that there could the such a thing as "principal" eigenpairs,i.e, those which cannot be produced by any previous eigenfunction and consequently they would form the building blocks for any other eigenfunction that would ensue.

Discovering methods to do such a thing would be huge breakthrough that will bolster the prospects of the framework presented herein and might even end up effectively solving nonlinear control!

5.5 Testing the Framework on More Challenging Complex Systems

The examples presented in this thesis were very rudimentary as the purpose was more exploratory and expository. The aim was primarily to concretely establish the framework and ensure it's working on some low dimensional but nonlinear systems. A failure in the aforementioned systems would preclude attempts on the more complex systems pictured in Figures 5.4,5.5 and 5.6.

It is only natural that this framework be tested on complex, nonlinear, highly dynamic systems to assess its deployability in real time.

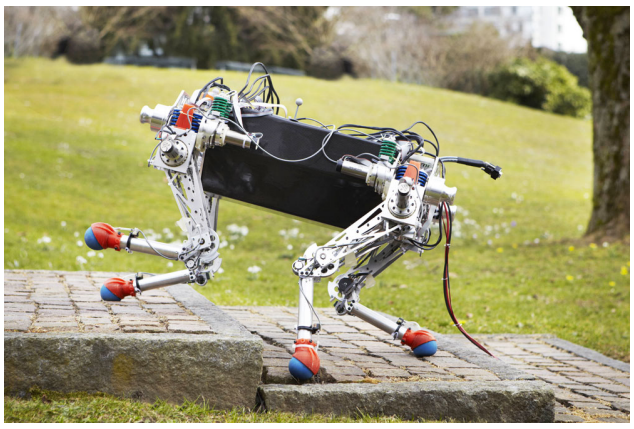

Figure 5.4: A Quadrupedal Robot

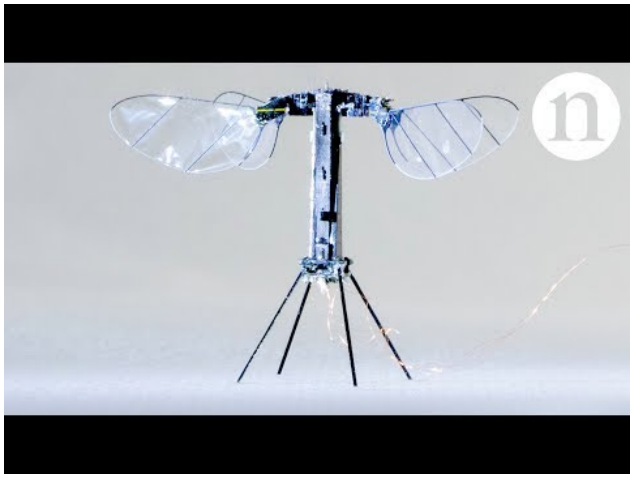

Figure 5.5: An Insect Inspired Robot

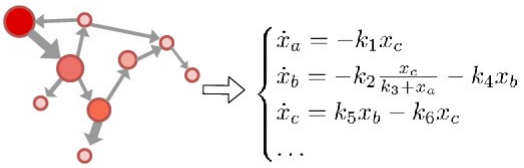

Figure 5.6: Dynamics of Biological Networks

BIBLIOGRAPHY

[1] 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.

[2] S. L. Brunton, B.W. Brunton, J. L. Proctor, E. Kaiser, and J. N. Kutz. Chaos as an intermittently forced linear system. *Nature Communications*, 8(19), 2017.

[3] S.L. Brunton, B.W. Brunton, Proctor J.L., and Kutz J.N. Koopman invariant subspaces and finite linear representations of nonlinear dynamical systems for control. *PLoS ONE*, 11(2), 2016.

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

[5] 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.

[6] 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. 10th IFAC Symposium on Nonlinear Control Systems NOLCOS 2016.

[7] Th. Buzug and G. Pfister. Optimal delay time and embedding dimension for delay-time coordinates by analysis of the global static and local dynamical behavior of strange attractors. *Phys. Rev. A*, 45:7073–7084, May 1992.

[8] Kathleen Champion, Bethany Lusch, J. Nathan Kutz, and Steven L. Brunton. Data-driven discovery of coordinates and governing equations. 116(45):22445–22451, 2019.

[9] Varad Deshmukh, Elizabeth Bradley, Joshua Garland, and James D. Meiss. Using curvature to select the time lag for delay reconstruction. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 30(6):063143, 2020.

[10] Andrew M. Fraser and Harry L. Swinney. Independent coordinates for strange attractors from mutual information. *Phys. Rev. A*, 33:1134–1140, Feb 1986.

[11] Craig Gin, Bethany Lusch, Steven L. Brunton, and J. Nathan Kutz. Deep learning models for global coordinate transformations that linearize pdes. *arXiv*, 2019.

[12] Eurika Kaiser, J. Nathan Kutz, and Steven L. Brunton. Data-driven discovery of Koopman eigenfunctions for control. In *APS Division of Fluid Dynamics Meeting Abstracts*, APS Meeting Abstracts, page M27.006, November 2017.

[13] B. O. Koopman. Hamiltonian systems and transformations in hilbert space. *Proceedings of the National Academy of Sciences*, 17(5):315–318, 1931.

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

[15] J. N. Kutz, S. H. Rudy, A. Alla, and S. L. Brunton. Data-driven discovery of governing physical laws and their parametric dependencies in engineering, physics and biology. In *2017 IEEE 7th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, pages 1–5, 2017.

[16] J. Nathan Kutz, Joshua L. Proctor, Steven L. Brunton, and Bingni W. Brunton. *Dyanamic Mode Decomposition: Data Driven Modeling of Complex Systems*. SIAM, 2016.

[17] Q. Li, F. Dietrich, E.M. Bollt, and I.G. Kevrekedis. 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), 2017.

[18] N. M. Mangan, S. L. Brunton, J. L. Proctor, and J. N. Kutz. Inferring biological networks by sparse identification of nonlinear dynamics. *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, 2(1):52–63, 2016.

[19] J.L. Proctor, S.L. Brunton, and J.N. Kutz. Generalizing koopman theory to allow for inputs and control. *Journal of Applied Dynamical Systems*, 17(1):909–930, 2018.

[20] 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.

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

[22] Floris Takens. *Detecting strange attractors in turbulence*, volume 898, page 366. 1981.

[23] M.O. Williams, I.G. Kevrekidis, and C.W. Rowley. A data-driven approximation of the koopman operator: Extending dynamic mode decomposition. *Journal of Nonlinear Science*, 25:1307–1346, 2015.

Appendix A

PROJECT FILES AND CODE

All the associated files and code can be found on the author's GitHub page in a repository linked below:

`https://github.com/AshwinSrinivas1994/EE700`

VITA

Ashwin Srinivas Badrinath earned two Master of Science degrees, one in Electrical Engineering and the other in Applied Mathematics from the University of Washington, Seattle. His master's thesis pertaining to data driven dynamical systems was advised by Prof. Sreeram Kannan and Prof. J. Nathan Kutz.

Prior to joining UW, he earned his bachelor's degree in Instrumentation Technology from R.V. College of Engineering located in Bangalore, India (2012-2016), following which he had two short stints working for Reliance Industries Ltd. as an engineering trainee in the instrumentation department and then as a project assistant at the Indian Institute of Science.

His research interest, at its core, lies in Control Theory and Signal Processing and their interplay and intersection with Machine Learning.