DATA-DRIVEN MODEL-FREE APPROXIMATION OF KOOPMAN OPERATOR FOR SYSTEM IDENTIFICATION USING DEEP LEARNING

Alexander Krolicki[*]**, Pierre-Yves Lavertu**[†]
Clemson University
Clemson, SC 29634 USA

ABSTRACT

Koopman spectral theory has provided a new perspective in the field of dynamical systems in recent years. Modern dynamical systems are increasingly non-linear and complex, and there is a need for a framework to model these systems in a compact and comprehensive representation which can be used for prediction and control. Koopman spectral theory represents nonlinear system dynamics in terms of a infinite-dimensional linear operator which acts on the space of all possible measurement functions of the system. Obtaining a linear representation of nonlinear dynamics allows engineers to leverage optimal control methods which have been rigorously developed in linear systems theory. The central problem in applying Koopman theory to a system of interest is that the choice of finite-dimensional basis functions is typically done apriori, using expert knowledge of the systems dynamics. Our approach learns these basis functions using a data-driven model-free approach where a deep neural network learns the basis functions which map from the finite-dimensional nonlinear space to the infinite-dimensional linear space. In order to identify a model of the system that is suitable for control, the learned basis functions are subject to constraints which ensure the trajectory of the transient dynamics of the system are accurately predicted such that the model is ideal for use in the design of a controller. We demonstrate this approach on a simple pendulum example in which we obtain a linear representation of the non-linear system and then predict the future state trajectories given some initial conditions. We also explore how changing the input representation of the time series dynamic system data can improve the quality of learned basis functions to provide a better approximation of transient system dynamics which otherwise may not be captured. This alternative representation is compared to the traditional raw time series data approach to determine which method results in lower reconstruction and prediction error of the true non-linear dynamics of the system.

1 Introduction

Dynamical systems are systems for which a function defines the time dependency of a point in a finite dimensional space. There are plenty of examples of such systems from airplanes flight paths to motion of a liquid in a container and while some can be described by sets of non-linear equations representing the desired physics, this approach can sometime be a complex challenge. Numerical modeling is a very appealing approach used in simulation which can help represent very complex non-linear systems though if very often lead to very demanding computation for the central processing units (CPUs) or simply put - they are CPU intensive simulations. This becomes a big challenge in controlled dynamic systems where the controller feedback to the dynamic system is time sensitive and lagging information can have critical consequences.
Over the past 2 decades or so, a new family of methods capable of efficiently approach the challenge posed by the complexity of certain dynamic systems have emerged: Data-driven modeling method supporting the characterization of

[*]Masters Student, *Department of Mechanical Engineering*, `akrolic@clemson.edu`
[†]PhD Student, *Department of Materials Science and Engineering*, `plavert@clemson.edu`

high-dimensional, non-linear dynamical systems. Though introduced in more details later, Dynamic mode decomposition (DMD) has been leading the pack in terms of promising techniques and more recently Deep learning has grabbed most of the research attention. All methods rely to some extent on the Koopman operator theory [1] also described later. In the case of dynamic systems, the data can be generalized as the knowledge of a variable in relation to the given system state. One good example could be the positions of flying leaves in the air at a time t. This type of observable can often be captured and stored in addition to any control inputs. Data-driven modeling methods. For DMD and neural network modelling techniques, data is at the epicenter of the approach.
DMD and deep learning approaches have already been used in many different areas with a substantial level of success and some of those examples are now presented.

1.1 Data-driven modeling method relevance through examples:

There is a broad spectrum of applications that can benefit from the Data-driven modeling methods though not all dynamical systems are well suited. As mentioned already, one important aspect of the deep neural network approach to learn the basis functions is that it is heavily dependent on access to existing data or ways to generate the required data. A good example and relevant application are building environment control system where large amounts of sensor information is captured and even virtual building simulation can be used to generate datasets. In this case, using Koopman operators facilitates the comparison of complex data while also simplifying the system representation. In Eisenhower et al. 2016 [2], the authors use the spectral decomposition approach to analyze building system data and state that the approach accelerates the comparison between models and data as well as draw conclusions about the sensor functions. In the end, using the spectral decomposition can help understand the changes in the different parts of a building and help highlight poor control performance.

The same approach can also be applied to power grid systems to help evaluate the response to continuously changing local demand. In this case, the challenge is in the size of the system and the amount of heterogeneous dynamic sub-systems like power plants, transmission lines and other renewable energy systems. The objective in using the Koopman operator technique here is to help identify key dynamic phenomena mechanisms as well as instabilities and help understand cascading power outages. In Susuki et al. 2017 [3], the authors list the following advantages to using Koopman Mode Decomposition (KMD): it creates a clear separation of spatio-temporal scales and enables direct computation from data without describing the underlying system.

Another application of Koopman operators is in autonomous vehicles driving technologies. Vehicle motion planning and control algorithms depend on the information of the vehicle dynamics. In this case, one big challenge is the creation of a trajectory planning and control module that can handle the complex, unstructured and unpredictable road scenarios. In Xiao et al. 2020 [4], the authors use Koopman operators but through a deep learning framework which is different from the EDMD (Extended Dynamic Mode Decomposition), ELM-EDMD (extreme learning machine-based EDMD), and MLP (multi-layer perception) methods. The authors list accuracy and efficiency as an advantage of their method and can be used in realistic scenarios in a CarSim environment.

Another Koopman operator application is in shared Control of Human-Machine Systems to help users accomplish tasks which otherwise would be challenging or even impossible for the user on his own. For this kind of system, the machine and human are working as partners where the machine works as an autonomous entity. One of the main challenges is that many different users may utilize the same mechanical device and requires a general solution enabling many different pair-up. In Broad et al. 2019 [5], the authors are using approximation of Koopman operator to learn the system dynamics through user interaction which apparently lead to the human-machine system spend a considerably superior amount of time in desirable states.

With the increasing interest in energy saving, one relevant application of the Koopman operator is in signalized traffic systems. With some work focusing on the usage of Intelligent Transport Systems (ITS) to help with the reduction of fuel consumption, one direct way to do so is in the usage of a model-free approach exploiting existing data to determine traffic signal control parameters to reduce queue lengths, traffic and potentially improving fuel consumption. In Ling et al. 2019 [6], the authors propose an approach using the Koopman operator theory to study signalized traffic flow networks. They demonstrate that the Koopman theory can be useful to detect queuing dynamics leading to instabilities for a variety of applications.

Data-driven modeling of nonlinear dynamic system has also potential in aerospace as a candidate to model the nonlinear dynamics fundamentals to the morphing airborne wind energy (AWE) aerostructures. In a nutshell, AWE would be use to extract energy from high-altitude winds through attached drones. According to Fonzi et al. 2020 [7], replacing the current rigid wings of drones with morphing structures would help produce more energy. The challenge lies in the fact that morphing wings are tightly coupled with the system aerodynamics and on top of that, AWE operation range is very broad. The complexity of that dynamic system makes it a very good candidate for the usage of DMD and

Koopman operator approach. In their study, Fonzi et al. [7] are able to improve the prediction performance over the existing modeling approaches.

1.2 Method Fundamental - The Koopman Operator

Bernard Koopman has pioneered the use of linear transformations on Hilbert space to analyze Hamiltonian systems by introducing the now-called Koopman operator and study its spectrum [1]. The Koopman operator is linear, infinite-dimensional and can be defined for any nonlinear dynamical system [8]. Though governing dynamics of a system can be finite-dimensional, the Koopman operator is infinite-dimensional, does not rely on linearization and captures the full information of the nonlinear dynamical system.
The following text is an extract of the section "Koopman Operator and Modal Decomposition" from Eisenhower et al. [2] as the description was found to be good. In order to define the properties of the Koopman operator, lets take a dynamical system evolution over a multi-dimensional finite space of the variables $x \in M$ which is defined as the state-space. The evolution in time of the variable x is described by the non-linear equation (1):

$$x\,(k+1)\ = f\,(x\,(k))\,, \tag{1}$$

where f is a function mapping the variables x at a given instant k to new variables forward in time at instant $k+1$. An observation function $M \rightarrow \mathbf{R}$ mapping the arbitrary variables on the manifold M to real numbers. The Koopman operator $K : R \rightarrow R$ is then defined as:

$$K\ g\,(x)\ = g\,(f\,(x))\,. \tag{2}$$

One can think of (2) as projecting the system dynamics into a different domain. The benefit of the Koopman operator is that it simplifies the initial non-linear evolution function f into an infinite-dimensional linear operator. It is important to mention that this process is done without discarding or losing any information of the initial dynamics.
Since K is a linear operator we can determine the eigenvalues and eigenfunctions associated with K,

$$K\psi_i\,(x) = \lambda_i \psi_i\,(x)\,, \tag{3}$$

for $i = 1, 2, 3, ...$ where $\psi i\,(x)$ is the $i-th$ eigenfunction corresponding to the $i-th$ eigenvalue λ_i .

As a final note, multiple approaches exist to calculate or approximate the Koopman modes and this work will focus on the usage of Neural Network inspired by the model developed in Han et al. 2020 [9]

1.3 Data-driven Models - Challenges:

One important requirement for a successful application of the Koopman operator theory is access to data. Indeed, most applications of the Koopman operator theory are considered data-centric which means that the ideal cases are where the data is already available. For some cases where the data might not be as easily accessible the data availability might become a big challenge. Data acquisition can be an expensive task both in time and dollars and as most companies are looking to reduce product development cycles and cost, the decision to invest in a data acquisition framework can be a difficult one to make. In retrospect, while the application of Koopman operator theory can help simplify the representation of highly complex dynamic systems, its implementation can be hindered by the data availability.
In line with data acquisition, the optimization of the sensor locations for data collection and decision-making is both a crucial and challenging task for any real-world application. The data quality is indeed highly dependent on the sensor location and the choice of what data to acquire can in the end impact the validity and accuracy of the model.

1.4 Convolutional/Deep Neural Network Hybrid Model:

The Koopman operator is used to represent non-linear dynamics in terms of an infinite-dimensional linear operator. The usage of this simplified linear operator can then be used for prediction, estimation, and control of complex non-linear systems. Dynamic mode decomposition (DMD) is considered the standard approach to approximate the Koopman operator from data. The DMD algorithm was initially developed and applied to fluid dynamic problems [10]. Because of its simple formulation and its direct application to measured data (experimental and simulation), it makes the DMD approach very compelling. The DMD algorithm uses the observation data to approximate both the eigenvalues and the Koopman modes without having to numerically implement a Laplace transform. DMD aims at finding the best approximation of Koopman operator on some finite-dimensional subspace and computing eigenfunctions of the resulting finite-dimensional linear operator. One of the main challenges with this approach though is that it can be very sensitive to noise in the data which can impact the quality of the Koopman operator approximation. Another challenge, or perhaps a limitation, it the interpolation of the model to different operating regime. Going back to the airborne wind energy application [7], the DMD approach becomes unstable when trying to interpolate between flying conditions. For that reason, improvements are brought to the DMD models to enable better interpolation between the flying regimes. Since the early adoption of the DMD model, there has been a lot of work to better understand as well as improve the model which led to multiple base model extension [7, 11–14]:

- Optimized DMD
- Optimal Mode Decomposition
- Exact DMD
- Sparsity Promoting DMD
- Multi-Resolution DMD
- Extended DMD
- DMD with Control
- Total Least Squares DMD
- Dynamic Distribution Decomposition
- Naturally Structured DMD

With the recent advancement in the machine learning field as well as the increase in computation power, there has been an increase in the usage of Deep Neural Network (DNN) for different applications including the learning of the Koopman operator. Indeed, recent research have started to look into the usage of DNN to learn the Koopman operator from the available data and the initial results. (Lusch et al., n.d.; Yeung et al., 2017) have investigated that alternate approach applied to autonomous settings while (Han et al., 2020) work focuses on the controlled dynamical systems. In this work, we focus on the later and try to investigate other potential Neural Network (NN) models to help improving the solution. More specifically, we implement a hybrid NN model using both CNN and DNN to improve the raw data usage and try to increase the accuracy of the solution as well as trying to limit the number of dimensions of the lifted space. Additionally, we are hoping that the data conversion into image type data could help deploy this model in cases where the data amount is low, as we believe it could enrich the input representation of the time series data.

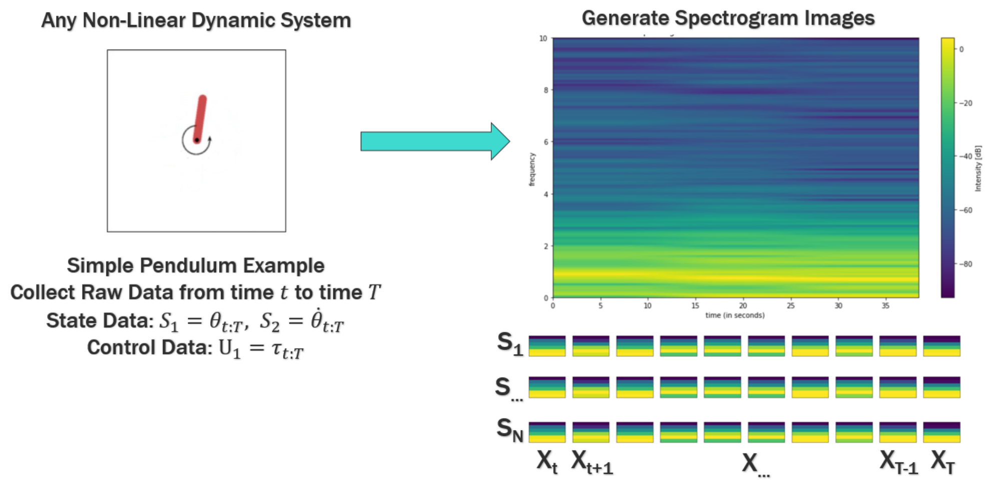
2 Algorithm

2.1 Overview:

Our implementation consists of several sub-processes which makeup the overall pipeline for determining the final linear lifted representation of the non-linear system. Provided time series data which contains measured state variables and control inputs, we first perform data pre-processing. During pre-processing, a Mel Spectrogram image is created by sampling the available time series dataset at a frequency less than the data was measured. In this case, we lose some of the resolution of the data, so in cases where high frequency sampling is not available this approach may not be as data efficient. Most modern engineering systems do have high sampling rates, so this is not a major concern but rather something to consider when applying it to the system of interest. Once the Spectrogram images are created, a convolutional autoencoder is trained to minimize the reconstruct error of the input image. At the output of the encoder in the convolutional autoencoder network, there is a bottleneck layer which captures the latent features of the images. These latent features are then parsed together with the raw time series data as inputs to a fully connected autoencoder network. Similarly, the autoencoder trains to minimize the reconstruction error of the enriched input representation. The bottleneck layer of this network is equal to the lifting dimension, and once the mean absolute error between the input and output of the network is minimized we can remove the decoder. Now we have lifted labels for the time series data generated by the fully connected encoder network which can be used to train our lifting DNN. Finally, the lifting DNN maps latent image data and raw time series data from the current time step to the lifted state representation for the next time step. The DNN lifting network is defined as our basis function which lifts our system from its original dimension to a truncated infinite dimension. In the end, we want to find a linear state space model which minimizes reconstruction error as well as prediction error compared to the real non-linear system. Another criteria for this system is that it be controllable. Once these 2 criteria are satisfied the linear system is evaluated to see how well it can track the state trajectories given some initial conditions and control inputs.

2.2 Data pre-processing:

After obtaining raw time-series data from the system of interest, data pre-processing steps must be taken in order to extract more useful information from each state in the system. The very first step is to create spectogram images from the raw time series data for all the system time. This step is done using the *specgram* function provided by ***matplotlib*** function library. Figure1 presents the raw time series data collection process and animated visualisation on the left and the spectogram representations of the raw data.



Any Non-Linear Dynamic System
Generate Spectrogram Images
Simple Pendulum Example
Collect Raw Data from time t to time T
State Data: $S_1 = \theta_{t:T}$, $S_2 = \dot{\theta}_{t:T}$
Control Data: $U_1 = \tau_{t:T}$
frequency
time (in seconds)
Intensity [dB]
S_1
$S_{...}$
S_N
X_t
X_{t+1}
$X_{...}$
X_{T-1}
X_T

Figure 1: Data Collection and Pre-Processing

Raw Input Data Representation:

Figure2 presents the entire state space and trajectories of all the data collected from the simple pendulum example. Several initial conditions are set during the collection of data from the system and results are stored for short durations of time. The control input to the system is randomly sampled from the action space. It is important to find a balance between the number of initial conditions you begin from and the duration of sampling data from the system. Note that for this approach to work well, a high sampling frequency of the state and control input must be used since it will be down=sampled once the spectrogram images are constructed.

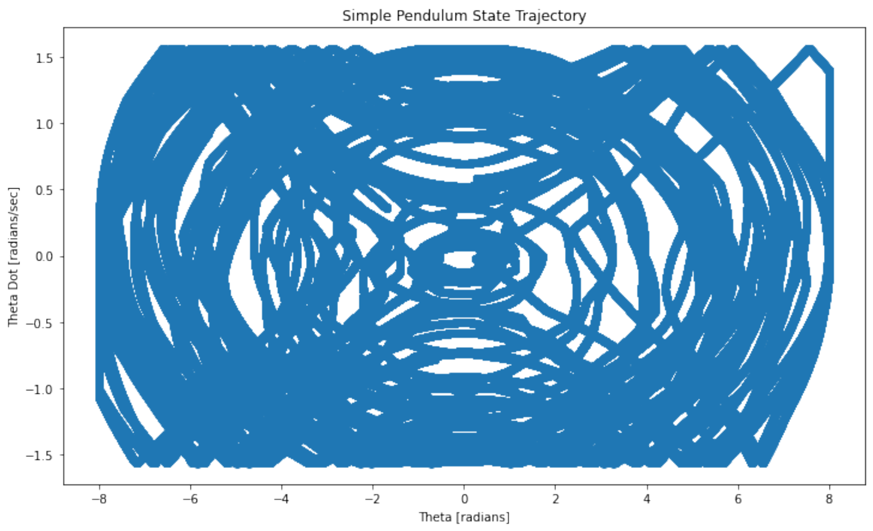


Figure 2: State space coverage and trajectories from collected raw data

Spectrogram Input Data Representation:

Once the spectogram images has been generated from the raw data, it is then gathered in a single image with the purpose of being used as the input to the initial convolutional autoencoder network (CAN). The image shown contains 10 intermediary time steps, and the entire image is representative as the raw state value sampled at the beginning of this interval. Every system requires that the spectrogram image be tuned in order to obtain a balance between the frequency resolution and the time resolution. The objective is to minimize the time step while retaining enough useful frequency information to distinguish between states.

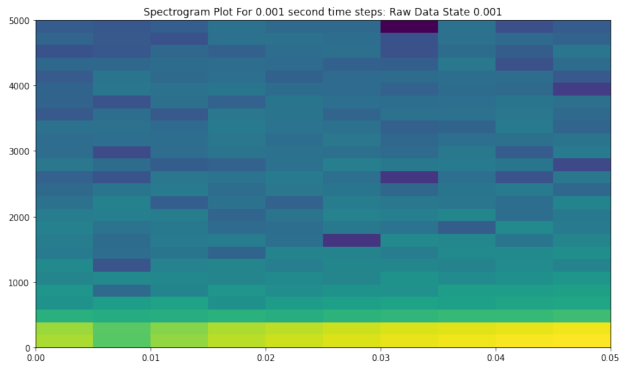


Figure 3: Spectrogram image used as input for the convolutional autoencoder network

Convolutional Neural Network Layers:

The purpose of the CAN shown in figure 4 is to learn the latent space enabling the reconstruction of the spectogram input. The encoder portion is composed of two convolution layers with LeakyReLU activation functions and including "same" padding. The decoder is also using LeakyReLU activation function with the exception of the last activation function which is Sigmoid. The main objective is to extract the latent representation of the state data as depicted in figure 5.

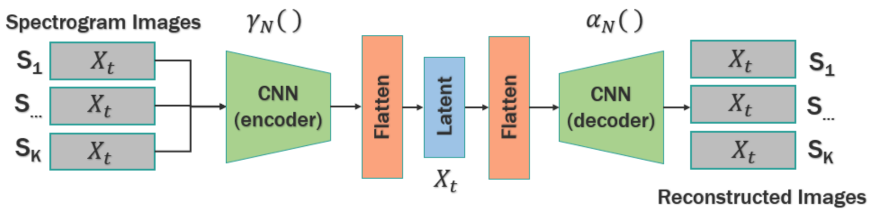


Figure 4: Convolutional autoencoder network

Encoder Network:

After training the encoder section shown in figure 5, the CAN generates the latent labels from the input images which are then combined with the raw state data.

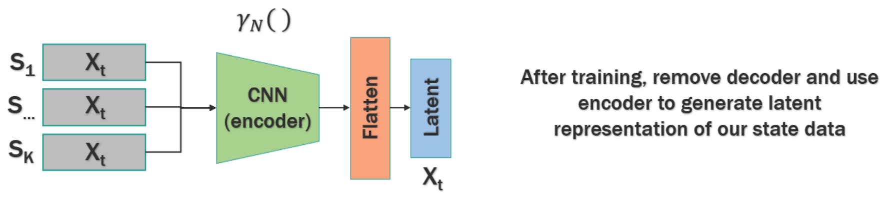

$\gamma_N()$
S_1
$S_{...}$
S_K
X_t
CNN (encoder)
Flatten
Latent
After training, remove decoder and use encoder to generate latent representation of our state data

Figure 5: Encoder network used to generate latent labels for each spectrogram image

Fully Connected Deep Neural Network Layers:

This section of the code as for main purpose to learn the lifted $N-th$ dimension linear representation able to predict the non-linear dynamics of the system. This portion is actually the core of the model-free approach for the learning of the Koopman operator. The encoder portion is composed of a single dense layer using the tanh activation function. Figure 6 summarize the overall big steps of second NN section. Figure 6 highlights the main objective with is to extract lifted basis functions.

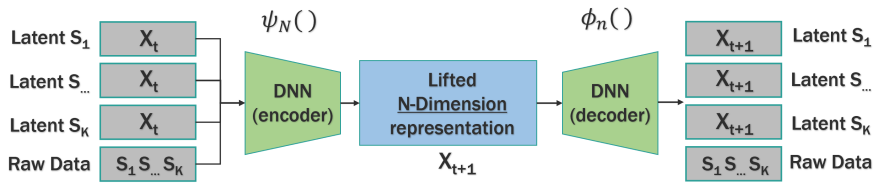


Figure 6: Fully connected autoencoder network

Lifted Representation of States:

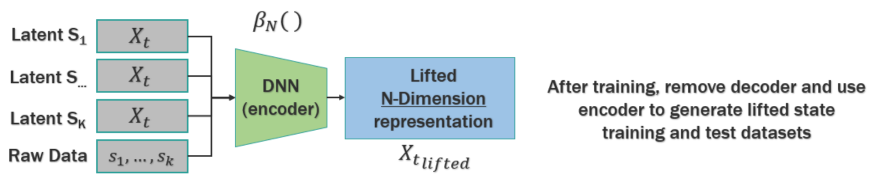

$\beta_N()$
Latent S_1
Latent $S_{...}$
Latent S_K
Raw Data
X_t
$s_1, ..., s_k$
DNN (encoder)
Lifted N-Dimension representation
$X_{t_{lifted}}$
After training, remove decoder and use encoder to generate lifted state training and test datasets

Figure 7: Fully connected encoder network (lifting basis function) mapping latent image data and raw time series data to the lifted N-dimensional states at the same time step.

Once the lifting basis functions $\psi i\,(x)$ have been learn, it can be used to predict the state space for $X_{t+1_{lift}}$ as shown in figure8.

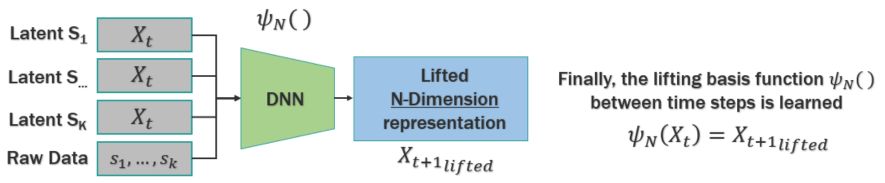

$\psi_N()$
Latent S_1
Latent $S_{...}$
Latent S_K
Raw Data
X_t
$s_1, ..., s_k$
DNN
Lifted N-Dimension representation
$X_{t+1_{lifted}}$
Finally, the lifting basis function $\psi_N()$ between time steps is learned
$\psi_N(X_t) = X_{t+1_{lifted}}$

Figure 8: Deep Neural Network that maps the latent image representations and raw data from the current time step to the lifted

Lifted Linear System Representation:

In our case, we are dealing with forced dynamical systems, which require a different set of equations to compute the lifted linear representation of the system. The derivation of these formulas used in this section are beyond the scope of this text, but are cited for reference [15].

In order to obtain the state space representation of our lifted linear forced dynamic system, we compute A, B, and C matrices where D is equal to zero.

$$[A, B] = \left[X_{t+1_{lift}} \begin{bmatrix} X_{tlift} \\ U_t \end{bmatrix}^T \right] \left[\begin{bmatrix} X_{tlift} \\ U_t \end{bmatrix} \begin{bmatrix} X_{tlift} \\ U_t \end{bmatrix}^T \right]^{-1} \tag{4}$$

$$C = X_t \left[X_{tlift} \right]^T \tag{5}$$

$$D = 0 \tag{6}$$

In order to evaluate the accuracy of the lifted system, we compute the loss term L_1 as defined in equation 7 which compares the labeled next lifted state to the linear dynamic systems output which should result in zero if the approximation is good.

$$L_1 = \ X_{t+1_{lift}} - [AX_{tlifted} + BU_t] \tag{7}$$

the controllability matrix Q is defined in equation 8, where the objective is to determine the number of linearly independent columns in Q by using the rank function. By definition, a linear dynamical system is controllable if the rank of the controllability matrix is equal to the lifted dimension in this case.

$$Q = \begin{bmatrix} B & AB & A^2B & \dots & A^{N-1}B \end{bmatrix} \tag{8}$$

Hence, we define our second loss function L_2 to be equal to the difference between the rank of the controllability matrix and the order of the system N.

$$L_2 = N_{lift} - rank(ctrb\,(A, B)) \tag{9}$$

So finally, the total loss L is computed in equation 10 as the sum of the lifted prediction error L_1 and the controllability criteria L_2. In order for a identified linear system to be accepted, it must have a total loss less than 0.005. Which would mean that the system is controllable as in equation 11 and accurate.

$$L = L_1 \ + \ L_2 \tag{10}$$

$$rank(Q) = N \tag{11}$$

3 Results

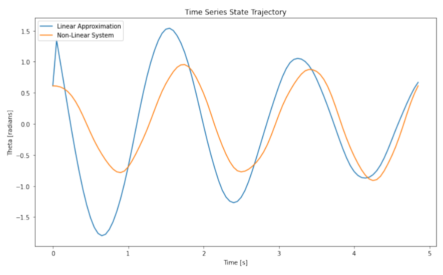
In this study, the usage of spectogram images from the time series dataset instead of raw time series data directly is evaluated to identify if the quality of the learned basis functions can be increased. Since the input representation includes images, the current neural network is composed of 2 sections: one CNN and one DNN. In the end, the objective remains the same with is to help identify the desired basis functions and the A and B matrices. One advantage of the current model is that it relies on very simple and well known CNN and DNN structures to help identify the latent space to then use in the next neural network section as described in figure 4 to 7. With the main objectives of increasing the quality of the learned basis functions that can accurately predict the transient dynamics for usage in the design of a controller we identified three criteria to evaluate the success of our designed model. The first criterion is the complexity of the model setup in terms of data gathering and preparation as well as the decoding of the model outputs for interpretation. A second criterion considered is the size of the A and B matrices as the size indicates how many dimensions are required by the model for what any user be considered an accurate representation which can be application dependent. For this second criterion, smaller A and B matrices are targeted as low dimensions models means that the number of lifting functions is lower and therefore would require a lower CPU cost. The last criterion in this study is the average state trajectory prediction of both θ and $\dot{\theta}$ as well as the error on the initial $\dot{\theta}$ prediction. Since the ultimate and ideal objective is to achieve good enough accuracy to use to generate a controller, it is important to get good prediction at the very start.

3.1 Criterion 1: Complexity of the model setup

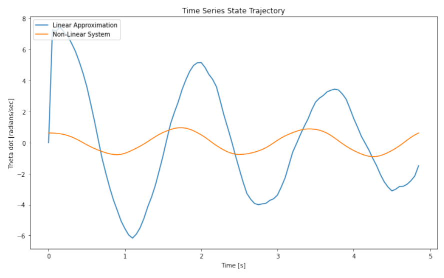
If we evaluate the model structure of different publications like Han et al. [9] and the example presented in Brunton et al. [16], they use DNN structures only which do not require to learn the different latent spaces. With only one type of model, the hyperparameters optimization study covers a smaller permutation spectrum as opposed to the current model proposed. Indeed, though the final results may already show improvements in the prediction quality, the path to fully master the structure is not a trivial one.
Leaving aside the neural network itself and focusing on the data pre-processing step, the simple fact that the proposed model uses spectrogram images means that the data needs an extra tranformation. The time series data conversion to image is a direct one but as the training progress through the network and reach the first DNN encoder, it then get merged with the initial raw data which requires some care in the structure to avoid any data corruption. For the first criteria, the proposed model bring extra complexity with the addition of the CNN part in comparison to the other models presented in the literature [9] [16].

3.2 Criterion 2: Size of the A and B matrices

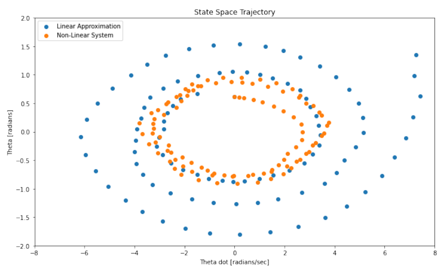
The way the code was implemented for this study gives the user the luxury of controlling the size of both A and B matrices to enable a direct comparison between the proposed approach and the model using the raw data only. The comparison here is done in terms of accuracy of the prediction and transient accuracy as well. As the goal is to head toward a controller, the better model would be able to achieve a accuracy within the design or implementation accuracy requirement for an extended period of time. The maximum lifted dimension used in this study was $N = 12$ and it was also the lifting dimension giving the best prediction in both case. In figure 9, it is possible to see that though the utilization of the latent images generates poor predictions for $t < 3$ it provides with good prediction after 3 seconds. The opposite trend can be observed on figure 10 in the case where only the raw data is used. Indeed, prediction preceding $t = 3$ are good while prediction after 3 seconds takes a less accurate path.



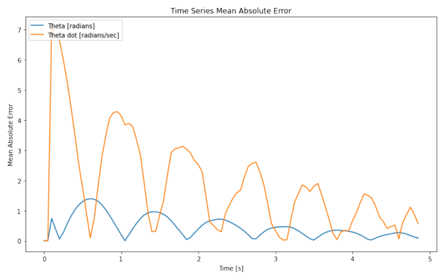
(a) Time series linear prediction for θ



(b) Time series linear prediction for $\dot{\theta}$

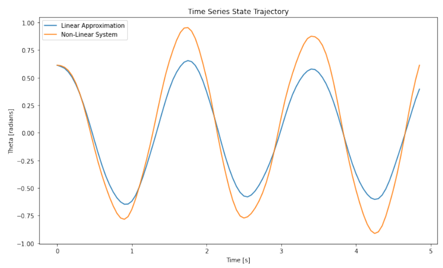


(c) State Space Trajectory Reconstruction

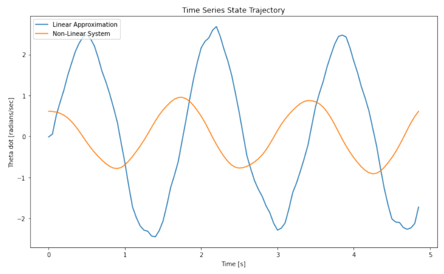


(d) Mean Absolute Error θ and $\dot{\theta}$

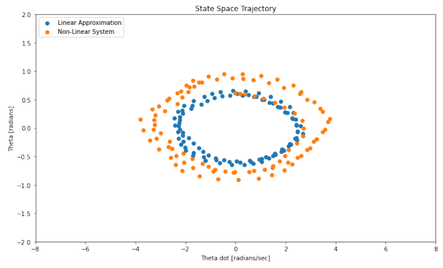
Figure 9: Raw data + Latent image data compared to non-linear system lifting dimension = 12



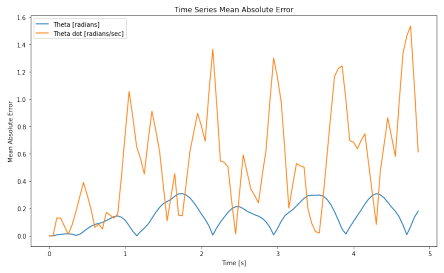
(a) Time series linear prediction for θ



(b) Time series linear prediction for $\dot{\theta}$



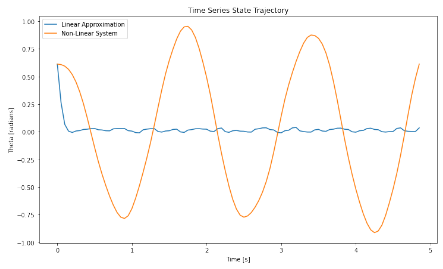
(c) State Space Trajectory Reconstruction



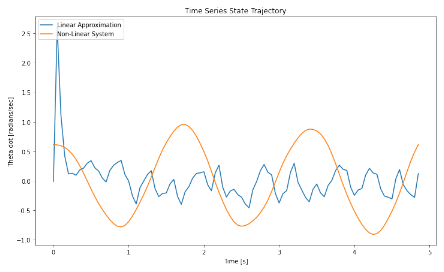
(d) Mean Absolute Error θ and $\dot{\theta}$

Figure 10: Raw data only compared to non-linear system lifting dimension = 12

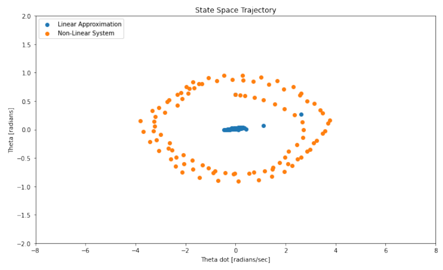
A close look at the prediction of θ for a lifting dimension $= 3$ when using the latent image and raw data highlight the poor performance of the predictions using the basis functions identified by the proposed model as shown in figure 11.



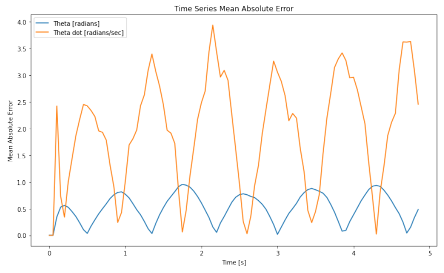
(a) Time series linear prediction for θ



(b) Time series linear prediction for $\dot{\theta}$



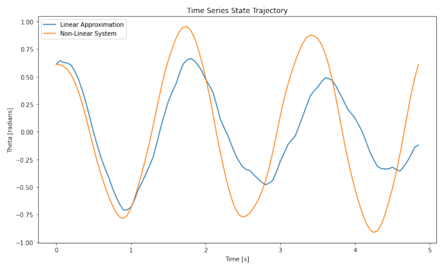
(c) State Space Trajectory Reconstruction



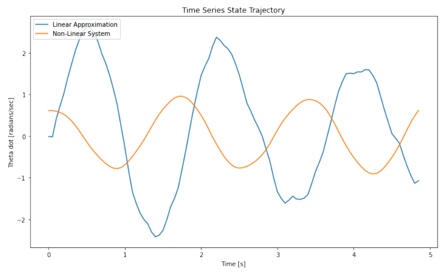
(d) Mean Absolute Error θ and $\dot{\theta}$

Figure 11: Raw data + Latent image data compared to non-linear system lifting dimension = 3

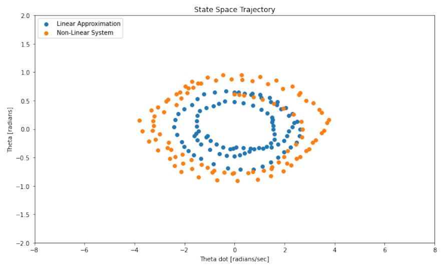
On the contrary, figure 12 shows that using the raw data only leads a model able to already predict accurately θ over for $t < 2$ before the model becomes inaccurate.



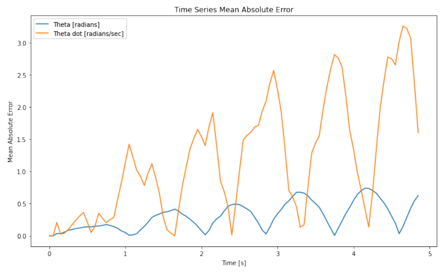
(a) Time series linear prediction for θ



(b) Time series linear prediction for $\dot{\theta}$



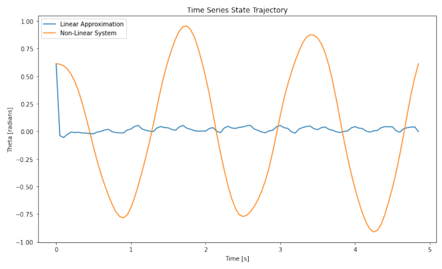
(c) State Space Trajectory Reconstruction



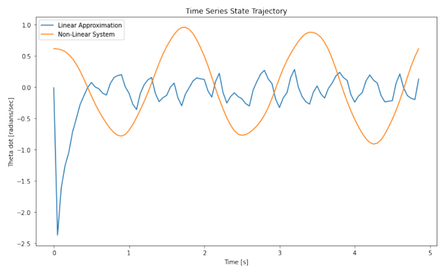
(d) Mean Absolute Error θ and $\dot{\theta}$

Figure 12: Raw data only compared to non-linear system lifting dimension = 3

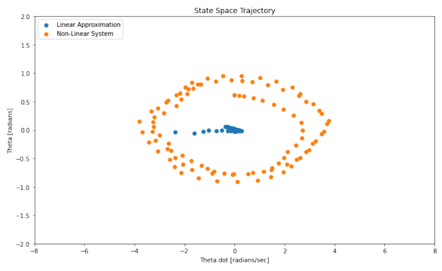
As the maximum learning dimension is increased to 5, the usage of latent image for the basis function identification again leads to poor predictive performance on all fronts (see figure 13)



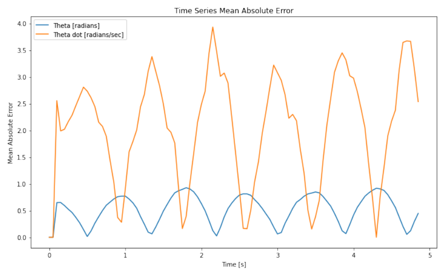
(a) Time series linear prediction for θ



(b) Time series linear prediction for $\dot{\theta}$



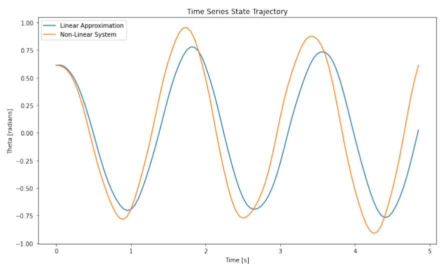
(c) State Space Trajectory Reconstruction



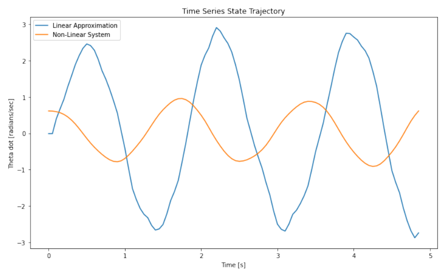
(d) Mean Absolute Error θ and $\dot{\theta}$

Figure 13: Raw data + Latent image data compared to non-linear system lifting dimension = 5

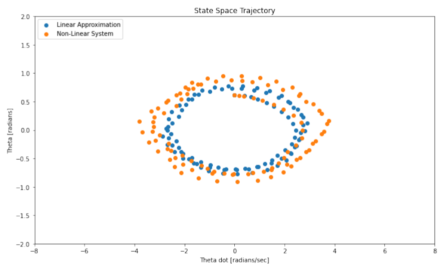
On the contrary, the prediction of the basis functions learned using only raw data are increasing in accuracy for both θ and $\dot{\theta}$ evolution through time (see figure 14).



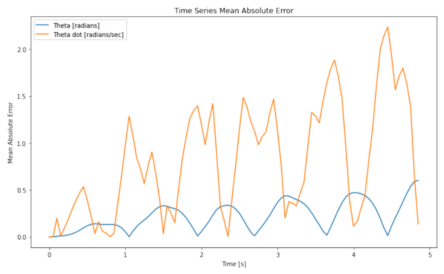
(a) Time series linear prediction for θ



(b) Time series linear prediction for $\dot{\theta}$



(c) State Space Trajectory Reconstruction



(d) Mean Absolute Error θ and $\dot{\theta}$

Figure 14: Raw data only compared to non-linear system lifting dimension = 5

For criterion 2, the current proposed model using latent image and raw data requires a higher lifting dimension for accurate prediction compared to a the basis functions identified using only the raw data. Indeed, using only raw data leads to accurate predictions.

3.3 Criterion 3: Average State Trajectory Prediction

For this last criteria, a comparison of the mean absolute error over average state trajectory prediction is presented in Table 1. The results show an improvement for the raw data only model between 3rd and 12th lifting dimension usage. The improvement trend only starts after the 5th dimension in the case where raw and latent image data is used for the basis function identification.

Average State Trajectory Prediction Mean Absolute Error				
Lifting Dimensions	$\theta*$	$\dot{\theta}*$	$\theta**$	$\dot{\theta}**$
3rd Dimension	0.515	1.95	0.30	1.18
5th Dimension	0.517	2.04	0.22	0.82
12th Dimension	0.456	1.84	0.14	0.54

Table 1: *Raw Data + Latent Image Data, **Raw Data Only

3.4 Potential Improvements

One potential downfall of the simple pendulum system is that it is almost too simple of a non-linear system for the spectrogram image to bear any meaningful transient frequency information. To highlight this, we show our spectrogram image from the pendulum in figure 15, its clear that there is little difference distinguishing the trajectory of the dynamics of the measured state since it is repeatedly cycling through the same set of fixed values. A more complex example of a nonlinear system could be the measurement of acoustics such as what is shown in figure 16. Visually it is clear that over time the state is evolving in a unique and identifiable manner, making the content of the image more useful for determining the trajectory of the audio or state. Ideally we would like to try out our approach on a more complex nonlinear system, especially a real system with noise and uncertainty to truly measure the robustness of the approach in performing accurate system identification.

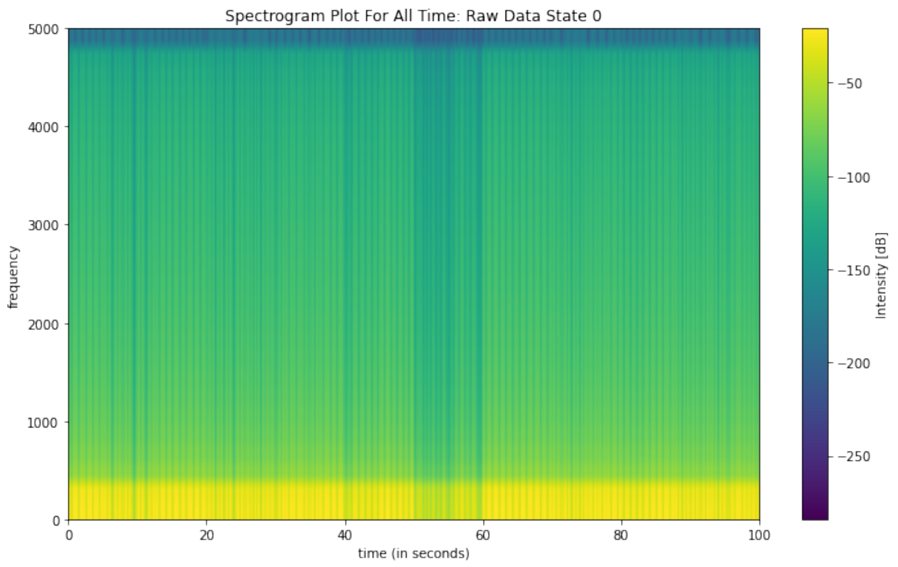


Figure 15: Periodicity of pendulum Spectrogram image for all training time

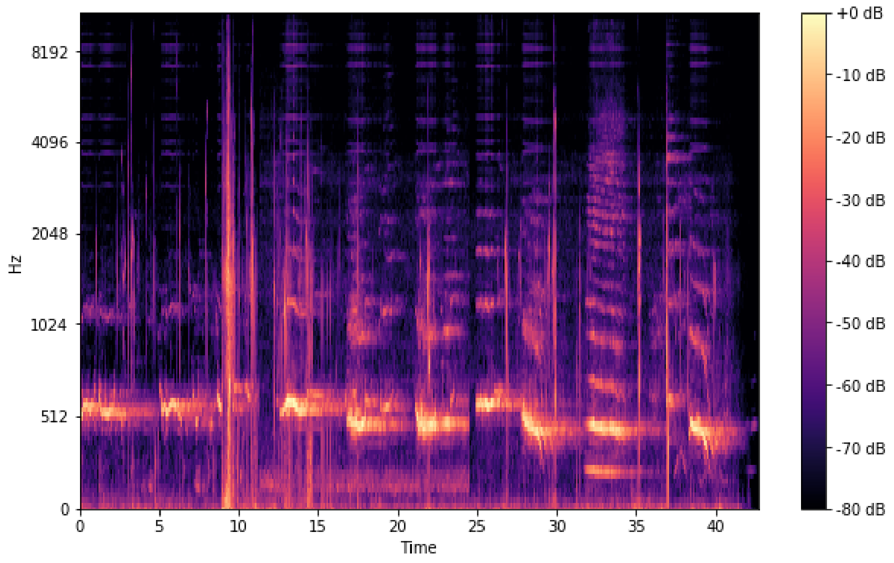


Figure 16: Spectrogram Image from acoustic data

4 Conclusion and Future Work

The investigation work around the utilization of spectogram images generated from a non-linear dynamic system state space time series data showed that the approach can lead to accurate predictions though it does not seem to be achieved as fast as when only using raw data. The current results provided by this work don't show conclusive evidence as to the benefit of the spectogram images in the quality increase in basis function identification at for low lifting dimensions.

Here, it worth mentioning that only a single Neural Network format was evaluated and it would be interesting to evaluate different activation functions and layer structure in both the CAN and DNN sections of the model. Also, the current model implementation uses a simple autoencoders and the usage of a VAE in the CNN portion of the code should be considered. While the spectogram images provide an enriched set of data, its accurate reconstruction through the initial autoencoder is crucial and could become a critical improvement leading to the actual desired outcome: An increased quality of the learned basis functions for more accurate prediction of transient dynamics.

A final aspect that wasn't consider in the current investigation is the actual level of non-linearity of the example system. The simple pendulum problem is considered as a simple non-linear problem and may be poorly suited to highlight the benefits of using spectogram image representation of the non-linear state space data. Now that the framework is available, it would be very relevant to test the proposed approach against another non-linear dynamic system like a walking quadruped or biped structure.

In conclusion, the current investigation showed that the usage of a latent image data representation increase the complexity of the Deep learning model though it doesn't present any increase in prediction accuracy until a lifting dimension of 12. At that dimension, the approach shows a good capture of the transient aspect of the dynamic while the usage of only raw data still shows better prediction over short time.

References

[1] Bernard Koopman. Hamiltonian systems and transformation in hilbert space. 1931.

[2] Bryan Eisenhower. Decomposing building system data for model validation and analysis using the koopman operator. *Fourth National Conference of IBPSA-USA*, 2010.

[3] Yoshihiko Susuki. Applied koopman operator theory for power systems technology. *Nonlinear Theory and Its Applications, IEICE*, 7:430–459, 2017.

[4] Yongqian Xiao. A deep learning framework based on koopman operator for data-driven modeling of vehicle dynamics. *ResearchGate*, 2020.

[5] Todd Murphey Brenna Argall Alexander Broad, Ian Abraham. Data-driven koopman operators for model-based shared control of human-machine systems. *The International Journal of Robotics Research*, 2020.

[6] Lillian J. Ratliff Samuel Coogan Esther Ling, Liyuan Zheng. Koopman operator applications in signalized traffic systems. 2019.

[7] U. Fasel N. Fonzi, S.L.Brunton. Data-driven nonlinear aeroelastic models of morphing wings for control. *royal society publishing*, 2021.

[8] Michael C. Mackey Andrzej Latosa. Studying chaos with densities. *Chaos, Fractals, and Noise*, 1994.

[9] Wenjian Hao Yiqiang Han. Deep learning of koopman representation for control. 2020.

[10] PETER SCHMID. Dynamic mode decomposition of numerical and experimental data. 2012.

[11] Umesh Vaidya Bowen Huang. Data-driven approximation of transfer operators: Naturally structured dynamic mode decomposition. 2017.

[12] Clarence W. Rowley Matthew O. Williams, Ioannis G. Kevrekidis. A data–driven approximation of the koopman operator: Extending dynamic mode decomposition. *Journal of Nonlinear Science*, 2015.

[13] Paul J. Goulart Andrew Wynn, Bharathram Ganapathisubramani. Optimal mode decomposition for unsteady flows. *Journal of Fluid Mechanics*, 2013.

[14] Clarence W. Rowley Kevin K. Chen, Jonathan H. Tu. Variants of dynamic mode decomposition: Boundary condition, koopman, and fourier analyses. *Journal of Nonlinear Science*, 2012.

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

[16] STEVEN L. BRUNTON. Modern koopman theory for dynamical systems. 2021.