Neural Koopman Lyapunov Control*

Vrushabh Zinage[1], Efstathios Bakolas[1]

Department of Aerospace Engineering and Engineering Mechanics, University of Texas at Austin, USA

Abstract

Learning and synthesizing stabilizing controllers for unknown nonlinear control systems is a challenging problem for real-world and industrial applications. Koopman operator theory allows one to analyze nonlinear systems through the lens of linear systems and nonlinear control systems through the lens of bilinear control systems. The key idea of these methods lies in the transformation of the coordinates of the nonlinear system into the Koopman observables, which are coordinates that allow the representation of the original system (control system) as a higher dimensional linear (bilinear control) system. However, for nonlinear control systems, the bilinear control model obtained by applying Koopman operator based learning methods is not necessarily stabilizable. Simultaneous identification of stabilizable lifted bilinear control systems as well as the associated Koopman observables is still an open problem. In this paper, we propose a framework to construct these stabilizable bilinear models and identify its associated observables from data by simultaneously learning a bilinear Koopman embedding for the underlying unknown control affine nonlinear system as well as a Control Lyapunov Function (CLF) for the Koopman based bilinear model using a learner and falsifier. Our proposed approach thereby provides provable guarantees of asymptotic stability for the Koopman based representation of the unknown control affine nonlinear control system as a bilinear system. Numerical

*This research has been supported in part by NSF award CMMI-1937957

[1]Graduate student at the Department of Aerospace Engineering and Engineering Mechanics, University of Texas at Austin, Texas, USA

[2]Associate Professor at the Department of Aerospace Engineering and Engineering Mechanics, University of Texas at Austin, Texas, USA

simulations are provided to validate the efficacy of our proposed class of stabilizing feedback controllers for unknown control-affine nonlinear systems.

Keywords: Koopman operator theory, Neural networks, Control Lyapunov Functions

2010 MSC: 00-01, 99-00

---

1. Introduction

Recently, Koopman operator techniques have proven to be powerful tools for the analysis and control of nonlinear systems whose dynamics may not be known a priori. The key idea of such methods is to associate a nonlinear system (nonlinear control system) with a linear system (bilinear control system) of higher dimension than the original system. The higher dimensional state spaces of these "lifted" linear or bilinear control systems are spanned by functions of states known as *Koopman observables*. Unlike linearization techniques, Koopman operator methods provide higher dimensional (lifted) linear or bilinear state space models which explicitly account for nonlinearities in the original system dynamics and their validity is not limited to a small neighborhood around a reference point or trajectory as in standard linearization approaches. However, since the Koopman operator is infinite-dimensional, the resulting lifted state space models will be also infinite-dimensional and consequently, the control design can become a complex, if not computationally intractable, task. In order to improve computational tractability, recent approaches in the field aim at characterizing a finite approximation of the Koopman operator via data-driven methods such as the Extended Dynamic Mode Decomposition (EDMD). EDMD mainly uses time series data to form a higher dimensional linear (bilinear) state space model that approximates the unknown nonlinear system (control-affine nonlinear control system). In addition, the connection of EDMD with Koopman operator theory has been explored in [1] and extended to non-sequential time series data in [2]. Further [3] shows the convergence in the strong operator topology of the Koopman operator computed via EDMD [4] to the actual

Koopman operator as the number of data points and the number of observables tend to infinity. Koopman operator theory (KOT) has been applied to robotics applications [5, 6, 7, 8, 9], power grid stabilization [10, 11], state estimation [12, 13], control synthesis [14, 15, 16, 17], actuator and sensor placement [18], aerospace applications [19, 20], analysis of climate, fluid mechanics and control of PDEs, to name but a few. Koopman operator theory postulates that a nonlinear uncontrolled system can be lifted to an equivalent (infinite-dimensional) linear system whereas a nonlinear control system to a bilinear control system. The major challenge in realizing this lifting process is that the Koopman observables are unknown and their characterization can be a complex task. [21] presents a deep learning framework for learning the Koopman observables for uncontrolled dynamical systems. However, the learned Koopman operator is not guaranteed to be stable. [22] proposes a method to guarantee stability by learning a stable Koopman operator by utilizing a particular operator parameterization that ensures that the computed Koopman operator is Schur stable. However, the applicability of [22] is restricted to uncontrolled systems. Further, [23, 24, 25] propose data-driven approaches for identification of Koopman invariant subspaces whose applicability is, however, limited to uncontrolled nonlinear systems.

There has been a wide interest in control design methods which are based on neural networks. Most of the proposed approaches in the relevant literature use reinforcement learning [26]. However, there are no theoretical guarantees that the designed control system is stabilizable [3] which is crucial especially for safety critical applications. Towards this aim, the notion of Control Lyapunov Function (CLF) from control theory can play a vital role in characterizing the stability properties of nonlinear control systems and designing stabilizing controllers (that is, controllers that guarantee closed-loop stability). CLFs were first studied by Sontag [27] and Artstein [28]. The existence of a CLF provides necessary and sufficient conditions for closed-loop stability of nonlinear control systems

[3] A system is stabilizable if there exists a feedback controller that can render the closed-loop system asymptotically stable

and can be viewed as a stability or safety certificate for such systems. In the control literature, there exist many approaches for the computation of Lyapunov functions for nonlinear control affine systems based on, for instance, sum-of-squares (SOS) and semidefinite programming (SDP) optimization [29, 30, 31, 32]. However, these methods do not typically scale well and their applicability is limited to polynomial control-affine systems. Further, [33] showed that for a particular dynamical system, there does not exist any polynomial Lyapunov function despite the dynamical system being globally asymptotically stable. Recent approaches [17, 34, 35, 36] use the notion of CLF to design a stabilizing MPC based controller for the Koopman based model of a nonlinear system. However, all these references assume that the CLF is a quadratic function (fixed structure). Using this fact, the CLF is computed by solving a convex optimization problem. Further, [17, 34, 35] assume that the Koopman based observables which are utilized for the computation of the approximation of a nonlinear system are known or guessed.

Finding a Lyapunov function for a nonlinear system is in general a challenging task. Recently, the so-called Lyapunov neural networks methods have been proposed to learn a valid Lyapunov function that will guarantee closed-loop stability of nonlinear systems. The candidate Lyapunov functions are parameterized by means of feedforward neural networks and the Lyapunov conditions are imposed as soft constraints in the learning (optimization) problem. These methods are motivated by the fact that any continuous function can be approximated by means of a neural network with a finite number of parameters that must be learned [37, 38]. A continuously differentiable function corresponds to a Lyapunov function for a nonlinear system if it satisfies certain conditions, which we refer to as Lyapunov conditions. One can verify whether a function learned by a neural network satisfies these conditions or not by utilizing techniques that can check certain properties of the outputs of the neural network. These verification techniques can be broadly classified into two main methods, one in which the verification is inexact and is carried out by solving a relaxed convex problem and another one in which the verification is exact and based on Mixed

Integer Programming (MIP) solvers and Sequential Modulo Theories (SMT) solvers. In [39, 40, 41, 42] Lyapunov control methods are proposed based on Lyapunov neural networks. [43] proposes formal synthesis methods for learning Lyapunov functions. However, the approaches proposed in [39, 40, 41, 42, 43] assume that the nonlinear dynamics are known. [44] proposes a framework for learning a Lyapunov function where the dynamics are not known. [45] provides stability certificates via a learned Lyapunov function using trajectory data only. [46] proposes a framework for discrete-time polynomial (nonlinear) systems and learns a safe region of attraction (ROA) using neural networks. However, these approaches are restricted to learning linear feedback controllers or neural network based feedback controllers and do not guarantee the existence of a stabilizing feedback controller or propose a systematic method to characterize it. Further, there are no tools to analyze the stability properties of unknown control-affine nonlinear systems. In contrast with the aforementioned references, in this work we utilize the Koopman operator theory framework to describe, analyze and control the behavior of any known or unknown control-affine nonlinear system via a higher dimensional (lifted) bilinear control system.

Contributions: The main contribution of our work is four-fold. 1) We propose a deep learning framework for simultaneously learning a stabilizable bilinear (lifted) state space model and the Koopman observables from data obtained from the open-loop trajectories of the latter system generated by random control inputs applied to the unknown control-affine nonlinear system. In our approach, closed-loop stability is guaranteed when our method can successfully learn a Control Lyapunov Function (CLF). Finding a valid CLF for a general nonlinear system is still considered an open problem. To that end, the key contribution of this work toward control design is a systematic method to compute a valid CLF using deep neural networks and verification algorithms. 2) Based on the learned CLF, we design a feedback controller using the celebrated universal Sontag's formula [27] that guarantees closed-loop asymptotic stability. 3) Our approach allows us to utilize state-of-the-art tools for verification based on SMT solvers even for the case in which the

nonlinear dynamics is unknown by computing a data-driven (lifted) bilinear system (approximation of the unknown system) via KOT based methods. 4) In contrast to recent methods [39, 40, 41, 42, 43, 44], which either restrict themselves to the class of linear feedback controllers or learn nonlinear feedback controllers represented by neural networks which do not offer guarantees of closed-loop stability in general, our method ensures that the computed feedback controller is a stabilizing one which can asymptotically steer the system to the origin. To the best knowledge of the authors, this is the first paper that proposes a method that simultaneously learns the observables together with a stabilizable Koopman Bilinear Form (KBF) which allows us to design stabilizing feedback controllers for the Koopman bilinear system. To illustrate and also validate the ability of our proposed class of stabilizing feedback controllers to steer nonlinear systems with unknown dynamics to the desired final state, we present numerical experiments for a nonlinear control system used in practical applications as well as a popular academic example.

Organization: The rest of the paper is organized as follows. Section 2 introduces the preliminaries followed by the problem statement. Section 3 discusses our approach to solve the problem. Section 4 discusses the results followed by some concluding remarks in Section 5.

2. Preliminaries and Problem Statement

Consider a control-affine nonlinear system given by

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}) + g(\boldsymbol{x})\boldsymbol{u} = f(\boldsymbol{x}) + \sum_{i=1}^{m} g_i(\boldsymbol{x})u_i, \quad \boldsymbol{x}(0) = \boldsymbol{x}_0 \tag{1}$$

where $f : \mathbb{R}^n \to \mathbb{R}^n$, $g = [g_1,\ g_2 \dots g_m]$, where $g_i : \mathbb{R}^n \to \mathbb{R}^n$ for all $i \in \{1, \dots, m\}$, $\boldsymbol{x} \in \mathcal{X} \subset \mathbb{R}^n$ is the state of the system and $\boldsymbol{u} = [u_1,\ u_2, \dots, u_m]^{\mathrm{T}} \in \mathcal{U} \subset \mathbb{R}^m$ is the control input applied to the system where $\mathcal{X}$ and $\mathcal{U}$ are compact sets (the assumption on compactness of $\mathcal{X}$ and $\mathcal{U}$ is made for learning purposes given that it would be practically impossible to solve the learning problem over unbounded admissible input or state sets; the reader should not perceive this

assumption as an indication of studying problems with state or input constraints). The function f is commonly known as the drift term (or vector field) whereas g is known as the actuation effect (or control vector field). We make the following assumptions:

Assumption 1. We assume that the function f is Lipschitz continuous whereas the function g is continuously differentiable.

Assumption 2. We assume that the origin $\boldsymbol{x} = 0$ is the unique equilibrium of the uncontrolled system $\dot{\boldsymbol{x}} = f(\boldsymbol{x})$ in $\mathcal{X}$ (in other words, $\boldsymbol{x} = 0$ is the unique solution to the equation: $f(\boldsymbol{x}) = 0$ in $\mathcal{X}$).

Further, in this paper, we consider the case that $f(\boldsymbol{x})$ and $g(\boldsymbol{x})$ are unknown in general. Next, we consider a discrete-time nonlinear control system which is obtained from the continuous-time system (1) after using a fourth order Runge-Kutta method:

$$\boldsymbol{x}_{k+1} = h(\boldsymbol{x}_k, \boldsymbol{u}_k), \quad \boldsymbol{x}_1(0) = \boldsymbol{x}_0 \tag{2}$$

where $\boldsymbol{x}_k \in \mathcal{X}$, $h : \mathcal{X} \to \mathbb{R}^n$. This discrete-time dynamical system will be used for construction of a dataset which would be used for training of the neural network (Section 3).

2.1. Koopman operator theory

In 1931, B. O. Koopman proved the existence of an infinite dimensional linear operator that can describe the evolution of functions of states of a nonlinear system, which are known as the observables [47]. Formally, let $\mathcal{F}$ be the space of functions spanned by the observables $\Phi : \mathbb{R}^n \to \mathbb{R}^N$, where $\Phi = [\phi_1,\ \phi_2, \ldots, \phi_N]^{\mathrm{T}}$ and $N > n$, then the Koopman operator $\mathcal{K} : \mathcal{F} \to \mathcal{F}$ is a linear infinite-dimensional operator that acts on functions $\Phi \in \mathcal{F}$ and is defined as follows:

$$\mathcal{K}[\Phi(\boldsymbol{x})] = \Phi \circ \mathcal{M}_t(\boldsymbol{x}), \tag{3}$$

where $\mathcal{M}_t$ denotes the flow map of the uncontrolled nonlinear dynamics $\dot{\boldsymbol{x}} = f(\boldsymbol{x})$ which is given by

$$\mathcal{M}_t\left(\boldsymbol{x}\left(t_0\right)\right) = \boldsymbol{x}\left(t_0\right) + \int_{t_0}^{t_0+t} f(\boldsymbol{x}(\tau))\mathrm{d}\tau. \tag{4}$$

It can be easily verified that $\mathcal{K}$ is a linear operator, that is, $\mathcal{K}[k_1\Phi_1 + k_2\Phi_2] = k_1\mathcal{K}[\Phi_1] + k_2\mathcal{K}[\Phi_2]$ for all $k_1, k_2 \in \mathbb{R}$ and $\Phi_1,\ \Phi_2 \in \mathcal{F}$. In practice, the Koopman operator is approximated by a finite-dimensional (truncated) operator which is subsequently used for modelling, analysis and control design. In this paper, we denote by $\tilde{\mathcal{K}}$ the finite-dimensional approximation (or truncation) of the Koopman operator $\mathcal{K}$ which can be represented by a matrix (we will use the same symbol for the latter operator and its matrix representation).

For control-affine nonlinear systems described by (1), the time derivative of Φ along the system trajectories is given by

$$\begin{aligned}\dot{\Phi}(\boldsymbol{x}) &= \nabla\Phi(\boldsymbol{x})[f(\boldsymbol{x}) + g(\boldsymbol{x})\boldsymbol{u}] = \nabla\Phi(\boldsymbol{x})f(\boldsymbol{x}) + \nabla\Phi(\boldsymbol{x})g(\boldsymbol{x})\boldsymbol{u} \\ &= \tilde{\mathcal{K}}\Phi(\boldsymbol{x}) + \nabla\Phi(\boldsymbol{x})\sum_{i=1}^{m} g_i(\boldsymbol{x})u_i. \end{aligned} \tag{5}$$

We assume that $\nabla\Phi(\boldsymbol{x})g_i(\boldsymbol{x})$ belongs to the span of $\Phi(\boldsymbol{x})$. In other words, there exists a constant matrix Q_i such that

$$\nabla\Phi(\boldsymbol{x})g_i(\boldsymbol{x}) = Q_i\Phi(\boldsymbol{x}).$$

This is a reasonable assumption to make given that, as is shown in the following lemma (which is taken from [48]), it holds true that for sufficiently large number of Koopman observables, the system governed by (1) can be equivalently modelled as a Koopman Bilinear Form (KBF):

Lemma 1. [48] For the system governed by (1) and a set of observables $\bar{\mathcal{Z}} = \{\boldsymbol{z}_i \in \mathcal{Z}\}_{i=1}^{\infty}$ which is a basis of $\mathcal{Z}$, where $\mathcal{Z} = \{h \in \mathcal{F} | h(\boldsymbol{x}_1, \boldsymbol{u}_1) = h(\boldsymbol{x}_2, \boldsymbol{u}_2)\}$, the Koopman based realization of the system (1) defined over $\bar{\mathcal{Z}}$ is bilinear.

Since $\boldsymbol{z}$ is a function of $\boldsymbol{x}$ only, therefore $\boldsymbol{\mathcal{Z}} := \{\boldsymbol{z} = \Phi(\boldsymbol{x}) : \boldsymbol{x} \in \mathcal{X}\}$ and Lemma 1 is applicable. The KBF form can then be written as follows [49]:

$$\dot{\boldsymbol{z}} = \tilde{\mathcal{K}}\boldsymbol{z} + \sum_{i=1}^{m} Q_i\boldsymbol{z}u_i, \tag{6}$$

where $\boldsymbol{z} := \Phi(\boldsymbol{x})$ and $\boldsymbol{z} \in \mathcal{Z}$ where $\mathcal{Z} = \{\boldsymbol{z} = \Phi(\boldsymbol{x}) : \boldsymbol{x} \in \mathcal{X}\}$. Note that $\boldsymbol{z} = 0$ is a equilibrium point for the bilinear system (6). After applying Euler discretization to the KBF and assuming zero order hold, we get

$$\boldsymbol{z}_{k+1} = \boldsymbol{z}_k + T\tilde{\mathcal{K}}\boldsymbol{z}_k + T\sum_{i=1}^{m} Q_i \boldsymbol{z}_k u_i = \tilde{\mathcal{K}}_d \boldsymbol{z}_k + \sum_{i=1}^{m} B_i \boldsymbol{z}_k u_i, \tag{7}$$

where T is the sampling period, $\tilde{\mathcal{K}}_d := I + T\tilde{\mathcal{K}}$, I is the identity matrix and $B_i = TQ_i$. Note that the local truncation error, $\|\boldsymbol{z}(t_k) - \boldsymbol{z}_k\|$, satisfies the following upper bound:

$$\|\boldsymbol{z}(t_k) - \boldsymbol{z}_k\| \leq pLT^2, \tag{8}$$

where L and p are given by

$$L := \left\|\tilde{\mathcal{K}}_d + \sum_{i=1}^{m} B_i \boldsymbol{z}_k u_i\right\|, \quad p := L \max_{kT \leq t < kT+T} \|\boldsymbol{z}(t)\|. \tag{9}$$

Note that the truncation error tends to zero as T tends to zero. Hence, even if for a given T, the truncation error between the states of the discrete-time system and the continuous-time system is not sufficiently small, T can be reduced accordingly. Note that in contrast to discretization approaches like Runge-Kutta, the Euler discretization preserves the bilinear form in the discrete-time state space model. This was the main motivation as to why Euler discretization was chosen over the Runge-Kutta methods.

Remark 1. Of particular interest is the case in which one can find a set of Koopman observables such that the unknown nonlinear system can be represented by a (continuous-time) linear time invariant (LTI) system as follows:

$$\dot{\boldsymbol{z}} = A\boldsymbol{z} + B\boldsymbol{u}. \tag{10}$$

The continuous-time LTI in (10) can be associated with the following discrete-time LTI system:

$$\boldsymbol{z}_{k+1} = A_d \boldsymbol{z}_k + B_d \boldsymbol{u}_k, \tag{11}$$

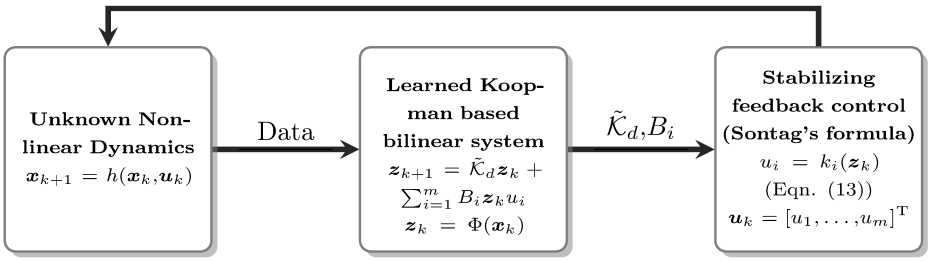


Figure 1: Learning and control framework for unknown nonlinear dynamics using Koopman operator theory

where $A_d = e^{AT}$, $B_d = \int_0^T e^{At} \mathrm{d}t B$ and T is the sampling period. Recently, the approach of lifting a nonlinear control system to a linear control system has gained a lot of attention [5] due to the availability of rich libraries of tools to analyze and design controllers for linear systems. However, lifting a nonlinear system into a higher dimensional linear system can be quite restrictive in practice because it may be hard to find a linear system that accurately describe the behavior of the original nonlinear system over a large portion of the state space, as pointed out in [50]. Further, the realization of the lifting process based on Koopman operator methods applied to a control-affine nonlinear system yields a bilinear control system rather than a linear system. This (lifted) bilinear representation of the control-affine nonlinear system has several advantages over the counterpart linear system representation as pointed out in [48].

2.2. Existence of stabilizing feedback controllers

Given the bilinear system (6), let us define the CLF as follows:

Definition 1. A Control Lyapunov Function is a continuously differentiable positive definite function $V : \mathcal{D} \to \mathbb{R}_+$, where $0 \in \mathcal{D}$ (that is, V is positive everywhere in $\mathcal{D}$ expect at $\boldsymbol{z} = 0$ where it is zero) such that the infimum of the

Lie derivative of V over all inputs is negative. More precisely,

$$\inf_{\boldsymbol{u}\in\mathcal{U}} \dot{V}(\boldsymbol{z}) < 0 \tag{12a}$$

$$\text{where} \quad \dot{V}(\boldsymbol{z}) := \frac{\partial V}{\partial \boldsymbol{z}}\dot{\boldsymbol{z}} = \frac{\partial V}{\partial \boldsymbol{z}}\tilde{\mathcal{K}}\boldsymbol{z} + \frac{\partial V}{\partial \boldsymbol{z}}\sum_{i=1}^{m} Q_i \boldsymbol{z} u_i. \tag{12b}$$

If the infimum of the Lie derivative of the CLF over all $\boldsymbol{u} \in \mathcal{U}$ is negative, then there exists a control input for which $\dot{V}(\boldsymbol{z})$ is negative along the trajectories of the closed-loop system. In particular, it can be shown that in the latter case there exists a feedback controller $\boldsymbol{u} := k(\boldsymbol{z}) = [k_1(\boldsymbol{z}), \ldots, k_m(\boldsymbol{z})]^{\mathrm{T}}$ that renders the closed-loop system asymptotically stable (in other words, $k(\boldsymbol{z})$ is a stabilizing feedback controller). This is stated formally as follows:

Theorem 1. [51] For the system given by (6), there exists a continuously differentiable function $k(\boldsymbol{z})$ for all $\boldsymbol{z} \in \mathbb{R}^N \setminus \{0\}$ and continuous at $\boldsymbol{z} = 0$ such that the controller $\boldsymbol{u} = k(\boldsymbol{z})$ renders the zero solution $\boldsymbol{z} = 0$ of the closed-loop system asymptotically stable if and only if there exists a Control Lyapunov Function (CLF) $V(\boldsymbol{z})$ such that

(C1) For all $\boldsymbol{z} \neq 0$, $\sum_{i=1}^{m} \frac{\partial V}{\partial \boldsymbol{z}} Q_i \boldsymbol{z} u_i = 0$ implies $\frac{\partial V}{\partial \boldsymbol{z}}\tilde{\mathcal{K}}\boldsymbol{z} < 0$

(C2) For all $\epsilon > 0$, there exists $\delta > 0$ such that $\|\boldsymbol{z}\| < \delta$ implies the existence of $\|\boldsymbol{u}\| < \epsilon$ satisfying (12)

The condition (C2) is also known as the small control property. If both conditions (C1) and (C2) hold true, then the feedback controller $\boldsymbol{u} := k(\boldsymbol{z}) = [k_1(\boldsymbol{z}), k_2(\boldsymbol{z}), \ldots, k_m(\boldsymbol{z})]^{\mathrm{T}}$, where the i-th component $k_i(\boldsymbol{z})$ of the feedback controller $k(\boldsymbol{z})$ satisfies the universal Sontag's formula [27]:

$$k_i(\boldsymbol{z}) = \begin{cases} -\frac{\boldsymbol{c}_i(\boldsymbol{z})[a(\boldsymbol{z}) + \sqrt{a^2(\boldsymbol{z}) + \sigma^2(\boldsymbol{z})}]}{\sigma(\boldsymbol{z})}, & \text{if } \sigma(\boldsymbol{z}) \neq 0 \\ 0, & \text{otherwise} \end{cases} \tag{13}$$

where $a(\boldsymbol{z}) = \frac{\partial V}{\partial \boldsymbol{z}}\tilde{\mathcal{K}}\boldsymbol{z}$, $\sigma(\boldsymbol{z}) = \sum_{i=1}^{m}(\frac{\partial V}{\partial \boldsymbol{z}} Q_i \boldsymbol{z} u_i)^2$, and $\boldsymbol{c}_i(\boldsymbol{z}) = \frac{\partial V}{\partial \boldsymbol{z}} Q_i \boldsymbol{z}$, will be a stabilizing controller (in other words, the controller $k(\boldsymbol{z})$ will render the closed-loop system asymptotically stable)

Remark 2. In contrast to recent methods which compute linear feedback controllers [39, 42] or nonlinear feedback controllers represented by neural networks [44] without offering any guarantees for closed-loop stability in general, our approach guarantees the existence of a stabilizing controller that will asymptotically steer the state of the Koopman based bilinear system (1) to the origin.

2.3. Problem statement

Next, we provide the precise formulation of the problem addressed in this paper:

Problem 1. Given the data snapshots X from the unknown nonlinear control system (1), compute the Koopman observables $\boldsymbol{z} = \Phi(\boldsymbol{x})$ and the matrices $[\tilde{\mathcal{K}}_d,\ B_1,\ \ldots,\ B_m]$ governing the Koopman based bilinear system (6). Further, design a feedback controller that renders the zero solution of the Koopman based bilinear system (6)

In the following sections, we explain in detail how our learning framework simultaneously learns the lifted bilinear system and a valid CLF, thereby consequently allowing us to design stabilizing feedback controllers for the lifted bilinear system.

3. Learning a stabilizable bilinear control system using Koopman operator theory

In this section, we present a learning approach to simultaneously learn the Koopman observables and a valid CLF for the learned bilinear system. Let the state snapshots $\{\boldsymbol{x}_k\}_{k=1}^{N_d}$ and the corresponding control inputs $\{\boldsymbol{u}_k\}_{k=0}^{N_d-1}$ be obtained when the control input $\boldsymbol{u}_k$ is applied to the discrete-time system (2) to transfer the state from $\boldsymbol{x}_k$ to $\boldsymbol{x}_{k+1}$ for all $k \in \{1, 2 \ldots, N_d - 1\}$ and let N_d denote the total number of data snapshots. Consider an encoder $\boldsymbol{z} = \Phi(\boldsymbol{x}; \theta) : \mathbb{R}^n \to \mathbb{R}^N$ which maps the state $\boldsymbol{x} \in \mathbb{R}^n$ to a higher dimensional lifted state $\boldsymbol{z} \in \mathbb{R}^N$ where $N > n$ and θ denotes the vector of parameters of the neural network. Similarly, let $\boldsymbol{x} = \Phi^{-1}(\boldsymbol{z}; \theta) : \mathbb{R}^N \to \mathbb{R}^n$ denote the decoder which maps the lifted state back to the original state $\boldsymbol{x}$ as shown in Fig. 2. For notational simplicity, we

represent $\Phi^{-1}(\boldsymbol{z};\theta)$ by $\Phi^{-1}(\boldsymbol{z})$. We construct a CLF $V(\boldsymbol{z};\theta)$, to be the output of a feedforward neural network. The main motivation for using feedforward neural networks for representing CLF is that they are expressive in the sense that any continuous function can be represented by means of a feedforward network with a finite number of parameters. For notational simplicity, we denote $V(\boldsymbol{z};\theta)$ by $V_\theta(\boldsymbol{z})$. Since the CLF has to be continuously differentiable, a smooth `tanh` activation function is used. One can also use the smooth approximations of the `ReLU` activation function. The objective is to simultaneously learn a valid CLF, Koopman observables Φ, the discrete-time Koopman operator $\tilde{\mathcal{K}}_d$ and the matrices B_i for all $i \in \{1, 2, \dots m\}$ that appear in the governing equations of the bilinear system by minimizing the following loss function $\mathcal{L}$ given by

$$\mathcal{L} = \alpha_1 \mathcal{L}_{\text{recons}} + \alpha_2 \mathcal{L}_{\text{dyn}} + \alpha_3 \mathcal{L}_{\text{lyap}} + \alpha_4 \mathcal{L}_{\text{ROA}} \tag{14}$$

where α_1, α_2, α_3 and α_4 are positive hyperparameters. Next, we describe the individual losses $\mathcal{L}_{\text{recons}}$, $\mathcal{L}_{\text{dyn}}$, $\mathcal{L}_{\text{phy}}$, $\mathcal{L}_{\text{lyap}}$, $\mathcal{L}_{\text{ROA}}$ whose weighted sum constitute the overall loss function $\mathcal{L}$.

i) **Reconstruction loss $\mathcal{L}_{\text{recons}}$**: The reconstruction loss ensures that the encoder is able to lift the state $\boldsymbol{x}$ and the decoder is able to project back the lifted state $\boldsymbol{z}$ to $\boldsymbol{x}$. The expression for $\mathcal{L}_{\text{recons}}$ is given by

$$\mathcal{L}_{\text{recons}} = \frac{1}{N_d} \sum_{k=1}^{N_d} \|\boldsymbol{x}_k - \Phi^{-1}(\Phi(\boldsymbol{x}_k))\|_2^2$$

ii) **Bilinear control system loss $\mathcal{L}_{\text{dyn}}$:** The dynamical system loss $\mathcal{L}_{\text{dyn}}$ (also known as the bilinear control system loss in this case) represents the extent to which the observables obey the governing Koopman based bilinear system and is given by the following expression

$$\mathcal{L}_{\text{dyn}} = \frac{1}{N_d - 1} \sum_{i=1}^{N_d - 1} \left\| \Phi\left(\boldsymbol{x}_{i+1}\right) - \tilde{\mathcal{K}}_d \Phi\left(\boldsymbol{x}_i\right) - \sum_{j=1}^{m} [\boldsymbol{u}_i]_j B_j \Phi\left(\boldsymbol{x}_i\right) \right\|_2^2$$

During every epoch, the matrices $\tilde{\mathcal{K}}_d, B_1, \dots, B_m$ are updated as follows:

$$\begin{bmatrix} \tilde{\mathcal{K}}_d & B_1 \dots & B_m \end{bmatrix} = \beta_{N_d} \Psi_{N_d}^T \left(\Psi_{N_d} \Psi_{N_d}^T \right)^{-1} \tag{15}$$

where the matrices Ψ_{N_d} and β_{N_d} are given by

$$\Psi_{N_d} = \left[\begin{pmatrix} 1 \\ \boldsymbol{u}_1 \end{pmatrix} \otimes \Phi(\boldsymbol{x}_1), \quad \cdots \quad , \begin{pmatrix} 1 \\ \boldsymbol{u}_{N_d-1} \end{pmatrix} \otimes \Phi(\boldsymbol{x}_{N_d-1}) \right],$$
$$\beta_{N_d} = [\Phi(\boldsymbol{x}_2), \ \ldots, \Phi(\boldsymbol{x}_{N_d})] \tag{16}$$

where $\otimes$ denotes the Kronecker product. Note that for given Φ and Φ^{-1}, the matrices $\tilde{\mathcal{K}}, B_1, \cdots B_m$ updated as in (15) minimize $\mathcal{L}_{\text{dyn}}$ if Ψ_{N_d} is a full row-rank matrix. This approach is known as the Extended Dynamic Mode Decomposition (EDMD) [2]. Note that Ψ_{N_d} can be made full row-rank if we increase the number of data-snapshots. Further, these matrices are updated optimally during every epoch of training the neural network. However, the major challenge is that the right Koopman observables Φ are unknown at the beginning. Hence, the loss initially is not zero and can actually be significantly large.

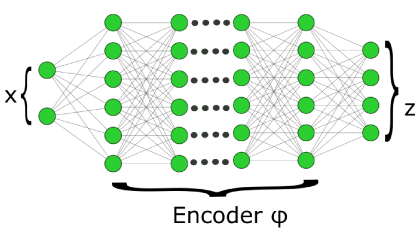
iii) **Control Lyapunov risk $\mathcal{L}_{\text{lyap}}$:** The control design based on CLF involves minimizing the minimax cost which is represented as follows [39]:

$$\min_{\theta, u \in \mathcal{U}} \max_{\boldsymbol{z} \in \mathcal{Z}} \left(\max\left(0, -V_\theta(\boldsymbol{z})\right) + \max\left(0, \nabla V_\theta(\boldsymbol{z}) \dot{\boldsymbol{z}}\right) + V_\theta^2(0) \right) \tag{17}$$

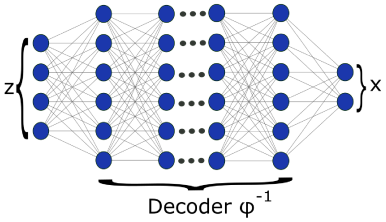
where $\nabla V_\theta(\boldsymbol{z})$ denotes the gradient of $V_\theta(\boldsymbol{z})$ with respect of $\boldsymbol{z}$. The time derivative of $V_\theta(\boldsymbol{z})$ along the system's trajectories is defined as follows:

$$\dot{V}_\theta(\boldsymbol{z}) := \nabla V_\theta(\boldsymbol{z}) \dot{\boldsymbol{z}} = \frac{\partial V}{\partial \boldsymbol{z}} \tilde{\mathcal{K}} \boldsymbol{z} + \frac{\partial V}{\partial \boldsymbol{z}} \sum_{i=1}^{m} Q_i \boldsymbol{z} u_i = \frac{\partial V}{\partial \boldsymbol{z}} \frac{(\tilde{\mathcal{K}}_d - I)\boldsymbol{z}}{T} + \frac{\partial V}{\partial \boldsymbol{z}} \sum_{i=1}^{m} \frac{B_i \boldsymbol{z} u_i}{T}$$

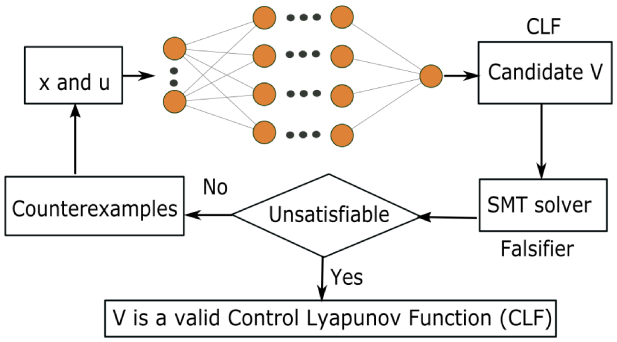
The first term in (17) ensures that the CLF is positive definite, the second term ensures that the Lie derivative of CLF is negative and the last terms ensure that the value of CLF at the origin is zero. The control Lyapunov loss function $\mathcal{L}_{\text{lyap}}$ measures the degree of violation of the Lyapunov conditions mentioned in (12). The common approach adopted in the relevant literature [39] is to utilize loss functions in order to transform the hard constraints on the CLF (12) into soft constraints (i.e., minimize $\mathcal{L}_{\text{lyap}}$). Let the value of $\min_{\theta, u \in \mathcal{U}} \max_{\boldsymbol{z} \in \mathcal{Z}} \left(\max\left(0, -V_\theta(\boldsymbol{z})\right) + \max\left(0, \nabla V_\theta(\boldsymbol{z}) \dot{\boldsymbol{z}}\right) + V_\theta^2(0) \right)$ be $G(\boldsymbol{z}^\star, \boldsymbol{u}^\star)$. If $V_\theta(\boldsymbol{z})$ is a valid CLF, then $G(\boldsymbol{z}^\star, \boldsymbol{u}^\star) = G(\boldsymbol{z}_1) = \cdots = G(\boldsymbol{z}_{N_d}) = 0$. Since



(a) An encoder $\Phi(\boldsymbol{x})$ which lifts the state $\boldsymbol{x}$ to a higher dimensional state space $\boldsymbol{z}$



(b) A decoder $\Phi^{-1}(\boldsymbol{z})$ which brings back the lifted state $\boldsymbol{z}$ to the original state $\boldsymbol{x}$



(c) Training a Lyapunov neural network to learn a Control Lyapunov Function (CLF) that ensures the existence of a stabilizing controller for the bilinear system (Theorem 1)

Figure 2: Deep learning framework to learn a stabilizable bilinear control system and a Control Lyapunov Function (CLF)

a valid CLF is not known during the training process and given the set $\mathcal{Z} = \{\boldsymbol{z}_1,\ \boldsymbol{z}_2, \ldots, \boldsymbol{z}_{N_d}\}$, the conditional probability $\mathbb{P}(G(\boldsymbol{z}^\star, \boldsymbol{u}^\star) = G(\boldsymbol{z}_1, \boldsymbol{u}_1)|\mathcal{Z}) = \mathbb{P}(G(\boldsymbol{z}^\star, \boldsymbol{u}^\star) = G(\boldsymbol{z}_2, \boldsymbol{u}_2)|\mathcal{Z}) = \cdots = \mathbb{P}(G(\boldsymbol{z}^\star, \boldsymbol{u}^\star) = G(\boldsymbol{z}_{N_d}, \boldsymbol{u}_{N_d})|\mathcal{Z})$. Therefore, the optimal unbiased Monte Carlo estimate [39] of the control Lyapunov risk is given by the sample mean as follows:

$$\mathcal{L}_{\text{lyap}} = \frac{1}{N_d} \sum_{i=1}^{N_d} \left(\max\left(0, -V_\theta(\boldsymbol{z}_i)\right) + \max\left(0, \nabla V_\theta(\boldsymbol{z}_i)\dot{\boldsymbol{z}}\right) + V_\theta^2(0) \right) \tag{18}$$

The hard constraint $V_\theta(0) = 0$ can be satisfied by manually setting the biases of the neural network $V_\theta(\boldsymbol{z})$ to be zero before the learning begins. Note that that setting the biases to zero is one way to ensure that the hard constraint $V_\theta(0) = 0$ is satisfied. An alternative way would be by choosing a candidate Control Lyapunov Function (CLF) as $V_\theta = \boldsymbol{z}^{\mathrm{T}} N_\theta^{\mathrm{T}} N_\theta \boldsymbol{z}$ where $N_\theta : \mathbb{R}^N \to \mathbb{R}^{N \times N}$ and then try to learn the function N_θ that characterizes V_θ. The violation of the Lyapunov conditions leads to failure in designing control inputs as these conditions need to be guaranteed over all states in $\mathcal{D}$. To avoid this issue, a first-order logic (also known as the falsification constraint) is incorporated which generates a counter-example that would not satisfy the Lyapunov conditions (12). In other words, the first-order logic $\mathcal{F}_\varepsilon(\boldsymbol{z})$ can be represented as

$$\mathcal{F}_\varepsilon(\boldsymbol{z}) = \Big(\sum_{i=1}^{N} \boldsymbol{z}_i^2 = \sum_{i=1}^{N} \Phi(\boldsymbol{x})_i^2 \geq \varepsilon > 0 \Big) \wedge \Big(V_\theta(\boldsymbol{z}) \leq 0 \vee \nabla V_\theta(\boldsymbol{z})\dot{\boldsymbol{z}} \geq 0 \Big) \tag{19}$$

To avoid numerical sensitivity issues, the value of ε is orders of magnitude smaller than the dimension of $\mathcal{Z}$. Further, ε is chosen such that $\sum_{i=1}^{n} \boldsymbol{x}_i^2 \geq \varepsilon$ would imply that $\sum_{i=1}^{N} \boldsymbol{z}_i^2 \geq \varepsilon$. Therefore, ε is chosen such that

$$\varepsilon \ll \min \left\{ 1,\ \|\mathcal{Z}\|,\ \min \left\{ \sum_{i=1}^{n} \boldsymbol{x}_i^2, \sum_{i=1}^{N} \boldsymbol{z}_i^2 \right\} \right\}. \tag{20}$$

We use a Satisfiability Modulo Theories (SMT) algorithm (SMT algorithms are used to determine whether a mathematical formula is satisfiable or not) for generating counterexamples which satisfy the falsification constraint (19). The problem of generating examples that satisfy the nonlinear constraints is highly non-convex and NP hard. However, recent progress among the class of SMT

algorithms has shown to be effective in solving problems with such nonlinear constraints. There are several SMT solvers, such as dReal [52], Z3 [53] and cvc5 [54], that one can utilize. However, in this work we will be using the dReal algorithm due to its $\delta-$completeness property [55] which is stated as follows:

Definition 2. [55] Let $\phi(\boldsymbol{x})$ be a quantifier first-order logic constraint. An algorithm is said to be $\delta-$complete if for any $\phi(\boldsymbol{x})$, the algorithm always returns one of the following answers correctly: ϕ does not have a solution (unsatisfiable), or there is a solution $\boldsymbol{x} = b$ that satisfies $\phi^{\delta}(b)$ where $\phi^{\delta}(b)$ is a small variation of the original constraint.

The neural network is trained until the SMT solver is not able to generate a counterexample satisfying the falsification constraint. Once a counterexample is generated, the training data are updated accordingly to further train the neural network. Note that a SMT solver never fails to generate counterexamples which satisfy the falsification constraint (19) if there are any. This is rigorously proved for SMT solvers such as dReal in [52].

If the dimension of the nonlinear system is large, learning a valid CLF using $\mathcal{L}_{\text{dyn}}$ and $\mathcal{F}_{\varepsilon}$ might be computational expensive for a SMT solver. To improve computational tractability, we simplify the computation of $\mathcal{L}_{\text{dyn}}$ and $\mathcal{F}_{\varepsilon}$ by considering the set of candidate CLF's as follows. Consider a set of candidate Control Lyapunov Functions $V(\boldsymbol{z};\theta)$ as follows:

$$V(\boldsymbol{z};\theta) = \boldsymbol{z}^{\mathrm{T}}(\gamma I + W_N(\boldsymbol{z})^{\mathrm{T}} W_N(\boldsymbol{z}))\boldsymbol{z}, \tag{21}$$

where $W_N(\boldsymbol{z})$ is a $n_w \times N$ matrix that corresponds to the output of a feedforward neural network, $\gamma > 0$ and n_w is the number of hyper-parameters. Clearly, $V(\boldsymbol{z};\theta)$ is positive definite given that the matrix $I + W_N(\boldsymbol{z})^{\mathrm{T}} W_N(\boldsymbol{z})$ is positive definite as the sum of a positive definite matrix and a positive semi-definite matrix.

Therefore, $\mathcal{L}_{\text{lyap}}$ and $\mathcal{F}_{\varepsilon}(\boldsymbol{z})$ can be written as follows:

$$\mathcal{L}_{\text{lyap}} = \frac{1}{N_d}\sum_{i=1}^{N_d}\max\left(0, \nabla V_\theta(\boldsymbol{z}_i)\dot{\boldsymbol{z}}_i\right),$$
$$\mathcal{F}_{\varepsilon}(\boldsymbol{z}) = \left(\sum_{i=1}^{N}\boldsymbol{z}_i^2 = \sum_{i=1}^{N}\Phi(\boldsymbol{x})_i^2 \geq \varepsilon\right) \wedge \left(\nabla V_\theta(\boldsymbol{z})\dot{\boldsymbol{z}} \geq 0\right)$$

Remark 3. A stabilizable Koopman based linear model can also be learned by suitably modifying $\mathcal{L}_{\text{dyn}}$ and $\mathcal{L}_{\text{lyap}}$ as

$$\mathcal{L}_{\text{dyn}} = \frac{1}{N_d - 1}\sum_{i=1}^{N_d-1}\left\|\Phi\left(\boldsymbol{x}_{i+1}\right) - \tilde{\mathcal{K}}_d\Phi\left(\boldsymbol{x}_i\right) - B_d\boldsymbol{u}_i\right\|_2^2$$

where the matrices A_d and B_d (Eqn. (11)) are updated during each epoch as follows:

$$[\tilde{\mathcal{K}}_d, B_d] \triangleq [A_d, B_d] = \beta_{N_d}\left[\Psi_{N_d}, \mathbf{U}\right]^{\dagger} \tag{22}$$

where † denotes the Moore-Pseudo inverse and β_{N_d}, Ψ_{N_d} and $\mathbf{U}$ are given by

$$\Psi_{N_d} = [\Phi(\boldsymbol{x}_1), \ldots, \Phi(\boldsymbol{x}_{N_d-1})], \tag{23a}$$
$$\beta_{N_d} = [\Phi(\boldsymbol{x}_2), \ldots, \Phi(\boldsymbol{x}_{N_d})], \tag{23b}$$
$$\mathbf{U} = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{N_d}]. \tag{23c}$$

Further the control Lyapunov risk and the falsification constraint remain the same except that $\nabla V_\theta(\boldsymbol{z})$ now becomes:

$$\nabla V_\theta(\boldsymbol{z}) = \frac{\partial V}{\partial \boldsymbol{z}}\frac{(\tilde{\mathcal{K}}_d - I)\boldsymbol{z}}{T} + \frac{\partial V}{\partial \boldsymbol{z}}B \tag{24}$$

iv) **Region of attraction (RoA) loss function $\mathcal{L}_{\text{ROA}}$:** Let $\phi(t, \boldsymbol{z})$ be the solution to the system of ordinary differential equations which describe the dynamics of a nonlinear system with initial condition $\boldsymbol{z}$ at time $t = 0$. Then, the Region of Attraction (RoA) is defined as the set of all points such that $\lim_{t\to\infty}\phi(t, \boldsymbol{z}) = 0$ [56].

Finding the exact region of attraction either analytically or numerically is not possible for many practical cases. However, one can use Lyapunov based methods

to estimate the RoA of nonlinear systems [57, 58]. If there exists a Lyapunov function that satisfies the conditions of asymptotic stability over a domain $\mathcal{Z}$, then the simplest estimate of RoA is given by the set $\Omega_c = \{\boldsymbol{z} \in \mathbb{R}^N : V(\boldsymbol{z}) \leq c\} \subset \mathcal{Z}$. The RoA loss $\mathcal{L}_{\text{ROA}}$ is defined as follows:

$$\mathcal{L}_{\text{ROA}} = \frac{1}{N_d} \sum_{i=1}^{N_d} \|\boldsymbol{z}_i\|_2 - \gamma_4 V_\theta(\boldsymbol{z}_i) \tag{25}$$

where $\gamma_4 > 0$ is a tunable parameter. The loss function $\mathcal{L}_{\text{ROA}}$ characterized how fast the CLF increases compared to the radius of the level sets. The region of attraction is also called as the region of asymptotic stability or domain / basin of attraction [56].

Remark 4. Although, the optimization problem (i.e., minimizing the loss given in (14)) is highly non-convex, recent results in deep learning have been successful in finding global minima for these non-convex problems.

The overall algorithm used to learn and control an unknown control-affine nonlinear system is described in Algorithm 1. Its main steps can be summarized as follows. The function `LEARNING` takes the data snapshots and returns the learned matrices governing the higher dimensional bilinear system, the Koopman observables (encoder), the decoder and a valid CLF. The learned CLF and the matrices governing the bilinear systems are then used to design a stabilizing feedback controller based on Sontag's formula (13). The function `CONTROL` takes the initial state $\boldsymbol{x}_0$ and computes the control sequence U (via the learned Koopman based bilinear model and the Control Lyapunov function) which steers the unknown nonlinear control system to the origin.

Algorithm 1 Learning Koopman operator based stabilizable bilinear model for control

Input: N_d, n, m, T, $X = \{\boldsymbol{x}_k,\ \boldsymbol{u}_k\}_{k=1}^{N_d}$

Output: Φ, Φ^{-1}, $\tilde{\mathcal{K}}_d$, $B_i\ \forall\ i \in \{1, \ldots, m\}$, $\boldsymbol{u} = k(\boldsymbol{z})$

function `LEARNING`($\{\boldsymbol{x}_k\}_{k=1}^{N_d},\ \{\boldsymbol{u}_k\}_{k=1}^{N_d}$)

  Repeat:

    $\Phi,\ \Phi^{-1} \leftarrow \mathrm{NN}_\theta(\boldsymbol{x}, \boldsymbol{z})$ ▷ Encoder and decoder

    $V_\theta(\boldsymbol{z}) \leftarrow \mathrm{NN}_\theta(\boldsymbol{z})$ ▷ Candidate Control Lyapunov Function (CLF)

    $\left[\tilde{\mathcal{K}}_d, \ldots, \mathrm{B}_m\right] \leftarrow \beta_{N_d} \Psi_{N_d}^T \left(\Psi_{N_d} \Psi_{N_d}^T\right)^{-1}$ ▷ Update bilinear system matrices

    $\nabla V_\theta(\boldsymbol{z}) \leftarrow \frac{\partial V}{\partial \boldsymbol{z}} \frac{(\tilde{\mathcal{K}}_d - I)\boldsymbol{z}}{T} + \frac{\partial V}{\partial \boldsymbol{z}} \sum_{i=1}^{m} \frac{B_i \boldsymbol{z} u_i}{T}$

    Compute total loss $\mathcal{L}(\theta)$ from Eqn. (14)

    $\theta \leftarrow \theta + \alpha \nabla_\theta \mathcal{L}(\theta)$ ▷ Update weights

  until convergence

  return $\Phi,\ \Phi^{-1},\ \left[\tilde{\mathcal{K}}_d, \ldots, \mathrm{B}_m\right],\ V_\theta(\boldsymbol{z})$

end function

function `FALSIFICATION`(X)

  Impose conditions defined in Eqn. (19)

  Use SMT solver to verify the conditions

  return satisfiability

end function

function `CONTROL`($\boldsymbol{x}_0$) ▷ Stabilizing feedback controller

  $\Phi,\ \Phi^{-1},\ \left[\tilde{\mathcal{K}}_d, \ldots, \mathrm{B}_m\right],\ V_\theta(\boldsymbol{z}) \leftarrow$ `LEARNING`(X)

  Repeat:

    $\boldsymbol{u} \leftarrow$ Eqn. (13), $U \leftarrow \mathrm{Append}(\boldsymbol{u})$ ▷ Universal Sontag's formula

    $\boldsymbol{x}_{\mathrm{next}} \leftarrow$ Apply feedback control law $\boldsymbol{u}$ to discrete system (Eqn. (2)) ▷ Propagate the state

    $\boldsymbol{x} \leftarrow \boldsymbol{x}_{\mathrm{next}}$

  until convergence

  return U

end function

function `MAIN`()

while Satisfiable **do**

    Add counterexamples to X

    $\Phi,\ \Phi^{-1},\ \left[\tilde{\mathcal{K}}_d, \ldots, \mathrm{B}_m\right],\ V_\theta(\boldsymbol{z}) \leftarrow$ `LEARNING`(X)

    $S \leftarrow$ `FALSIFICATION`(X)

end while

$U \leftarrow$ `CONTROL`($\boldsymbol{x}_0$)

end function

4. Results

In this section, we present results from numerical experiments which demonstrate the efficacy of our proposed learning -based stabilization approach on several nonlinear control systems. The learning framework is implemented in PyTorch for the simulations and our implementation follows Algorithm 1. The dReal package (SMT solver) is used to generate counterexamples for learning a valid CLF. In the following subsections, we first consider one popular academic nonlinear systems followed by one more realistic nonlinear systems.

4.1. Van der Pol oscillator

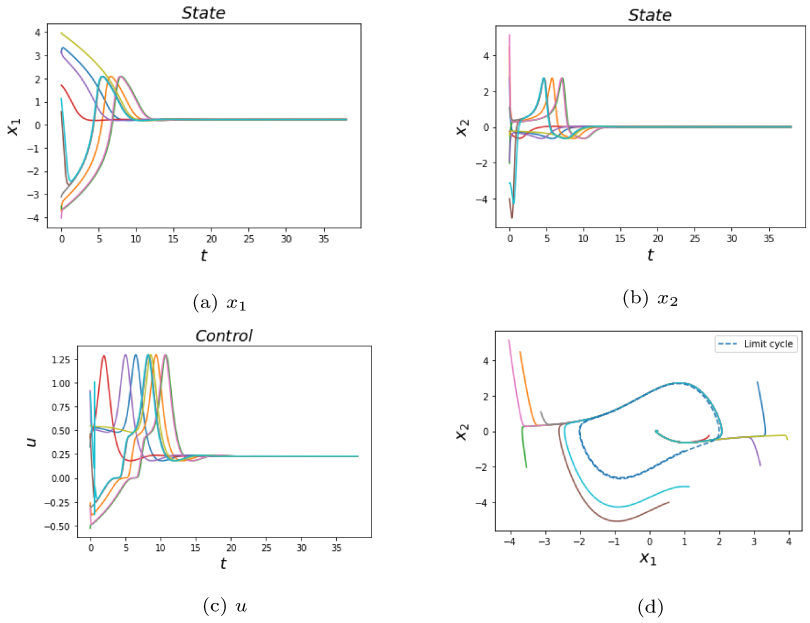
Next, we consider the Van der Pol oscillator system whose governing equations are given by

$$\dot{x}_1 = x_2, \qquad \dot{x}_2 = (1 - x_1^2)x_2 + x_1 + u \tag{26}$$

where $\boldsymbol{x} = [x_1,\ x_2]^{\mathrm{T}} \in \mathbb{R}^2$ is the state and $u \in \mathbb{R}$ is the control input for the Van der Pol oscillator. We define the state domain $\mathcal{X} = \{\boldsymbol{x} : \boldsymbol{x}_{\mathrm{lb}} \leq \boldsymbol{x} \leq \boldsymbol{x}_{\mathrm{ub}}\}$ (the inequality should be understood in the element wise sense) where $\boldsymbol{x}_{\mathrm{lb}} = [-10,\ -10]^{\mathrm{T}}$ and $\boldsymbol{x}_{\mathrm{ub}} = [10,\ 10]^{\mathrm{T}}$. We set the sampling time $T = 0.01$. The drift term and the control vector field of the Van der Pol system are polynomial and thus smooth. In addition, the system has an unique equilibrium. Therefore, both Assumptions 1 and 2 hold. Figures 5a and 5b show the evolution of the states of the Van der Pol oscillator with initial conditions $\boldsymbol{x}_0$ sampled from a uniform distribution over $\mathcal{X}$. Once a valid CLF is learned from training the neural network, it is used in the Sontag's formula to design a stabilizing feedback controller that would steer the state from $\boldsymbol{x}_0$ to the origin.

The control inputs are shown in Fig. 3c Note that the origin corresponds to the unstable equilibrium. The different solid colored curves in Figs. 5a, 5b and 3d represent the trajectories that originate from different initial conditions which are sampled from within $[-4, 4]^2$. Fig. 3c represent the control inputs which generated these trajectories. If no control input is applied, the trajectories will

converge to a limit cycle (shown by green dotted curve in Fig. 3d), irrespective of the initial state.



(a) x_1 (b) x_2 (c) u (d)

Figure 3: The evolution of the state and control input with time for the Van der Pol oscillator

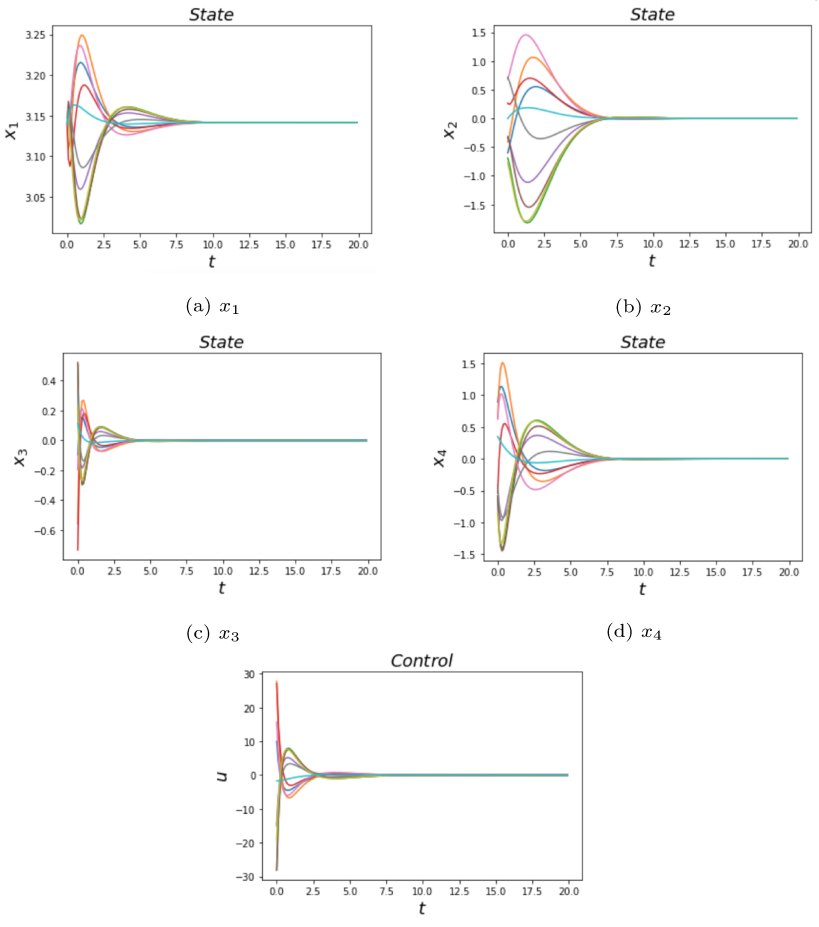
4.2. Cart pole system

The cart pole system is an underactuated system with one control input and two degrees of freedom (DOF). Due to its nonlinear structure, it is often used to validate nonlinear controllers. Cart pole systems can find many applications in, for instance, rocket propellers, self balancing robots, and stabilization of ships. The equations of motion of the cart pole system can be written as follows:

$$\begin{aligned} \dot{x} &= \frac{u + m_p \sin\theta \left(l\dot{\theta}^2 - g\cos\theta \right)}{m_c + m_p (\sin\theta)^2}, \\ \ddot{\theta} &= \frac{u\cos\theta + m_p l\dot{\theta}^2 \cos\theta \sin\theta - (m_c + m_p)\, g \sin\theta}{l\left(m_c + m_p(\sin\theta)^2\right)}, \end{aligned} \tag{27}$$

where $m_c = 4$kg is the mass of the cart, $l = 1$m and $m_p = 1$m are the length and mass of the pole respectively, $\boldsymbol{x} = [x_1,\ x_2,\ x_3,\ x_4]^{\mathrm{T}} = [x,\ \theta,\ \dot{x},\ \dot{\theta}]^{\mathrm{T}}$ is the state and u is the control input which controls the linear velocity of the cart.

Since the state space of the cartpole system is non-Euclidean (it is actually a manifold embedded in $\mathbb{R}^4$), we restrict our simulation-based analysis to a domain in which the system has a unique equilibrium. Further, both the drift term and the control vector field are smooth. Hence, both Assumptions 1 and 2 hold. The objective is to steer the initial state of the cart to an upright position. We set $T = 0.005$s. Fig. (4) shows the convergence of the trajectories starting from a set of ten randomly selected initial conditions to the origin when a stabilizing feedback controller is applied to the unknown control-affine nonlinear system.



(a) x_1

(b) x_2

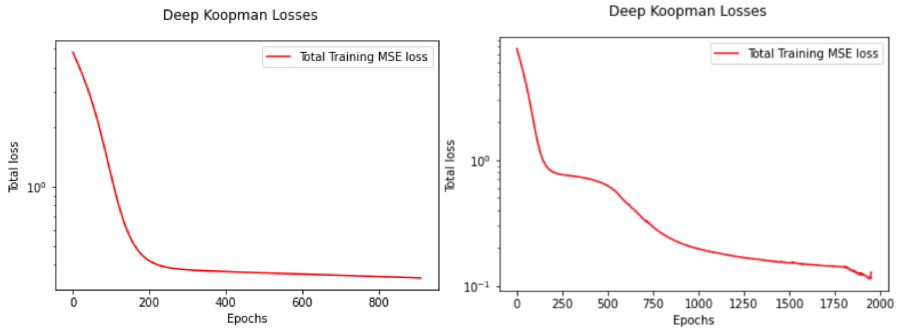
(c) x_3

(d) x_4

(e) u

Figure 4: The evolution of the state and control input with time for the cart pole system

The different solid colored curves in Figs. 4a-4d represent the trajectories that originate from different initial conditions. Fig. 4e illustrates the control inputs that generated these trajectories. Note that depending on the complexity of the system dynamics, the dimension of the lifted space N can be increased or decreased. Further, it was observed that the computational time required by the SMT solver to find counterexamples for the computation of a valid CLF increased exponentially with N. Fig. 5 shows the overall loss $\mathcal{L}$ for the Van der Pol oscillator and the cartpole system.



(a) Van der Polsystem

(b) Cartpole system

Figure 5: Total loss for the Van der Poland cartpole systems

4.3. Comparison with a recent method

We compare our approach with a recent method [17] which proposes a method to design stabilizing feedback controllers using the notion of CLF. In contrast with our approach, the authors of [17] assume that the CLF is quadratic and the Koopman based observables are guessed. Further, they assume that the nonlinear dynamics is known a priori. For our comparisons, we consider the Van der Pol oscillator and the cartpole example, where $N = 15$ is chosen for our approach. To implement the method proposed in [17], we choose monomials of degree 2 ($N = 15$) for the cartpole system and monomials of degree 4 ($N = 15$) for the Van der Pol oscillator. The metric used for our comparisons is the tracking error which is the Euclidean norm of the difference between the actual and desired states of the systems. It is evident from Fig. 1 that our approach

results in tracking errors with similar orders of magnitude for the Van der Pol oscillator. However, for the cartpole system, the approach [17] doesn't lead to the convergence of the states to the desired trajectory.

Tracking error	Method [17]	Our method
Van der Pol	7.95	6.58
Cartpole	9.12×10^5	8.35

Table 1: Comparisons with prior methods demonstrate the effectiveness of our approach

4.4. Discussion on numerical results and learning framework

The main advantage of using our proposed method over other standard approaches in the field [39, 40, 41, 42, 43, 44] is twofold. First, Koopman operator theory allows us to analyze an unknown control-affine nonlinear system via a learned higher dimensional Koopman based bilinear system. Second, unlike recent methods which restrict themselves to linear feedback controllers or learn a neural network based controller, our approach provides provable guarantees for closed-loop stability for the Koopman based bilinear system.

The training in our simulation study was performed for at least 900 epochs with Adam optimizer before the falsifier was used to generate counterexamples (that violate CLF conditions) which were then added to the training data. The activation function used for the encoder, decoder and the CLF is `Tanh` with learning rate set to 10^{-3} and the Mean Squared Error (MSE) loss for all the examples. The analytical expressions for $V_\theta(\boldsymbol{z})$ and $\Phi(\boldsymbol{x})$ are required to generate counterexamples by a SMT verifier (falsifier (19)) and compute the feedback stabilizing controller (Eqn. (13)). The expressions for $V_\theta(\boldsymbol{z})$ and $\Phi(\boldsymbol{x})$ are represented by the following recursive relations:

$$
\begin{aligned}
&\textbf{CLF:} \quad V_\theta(\boldsymbol{z}) = \texttt{tanh}(W^v_{h^v+1} y_{i+1} + b^v_{h+1}) \\
&\text{where} \quad y_i = \texttt{tanh}(W^v_i y_{i-1} + b^v_i), \;\; i = h^v, \;\; y^v_1 = \boldsymbol{z} \\
&\textbf{Encoder:} \quad \Phi(\boldsymbol{x}) = \texttt{tanh}(W^e_{h^e+1} y_{i+1} + b^e_{h^e+1}) \\
&\text{where} \quad y_i = \texttt{tanh}(W^e_i y_{i-1} + b^e_i), \;\; i = h^e, \;\; y^e_1 = \boldsymbol{x}
\end{aligned}
$$

where h^v, W_i^v, b_i^v and h^e, W_i^e, b_i^e denote the number of hidden layers, weights and biases of the CLF and encoder respectively. In order to train the encoder, decoder and the CLF, we assume that a black-box simulator of the unknown nonlinear dynamics (2) is available where $\boldsymbol{x}_{k+1}$ is returned given $\boldsymbol{x}_k$ and $\boldsymbol{u}_k$.

For the Van der Pol oscillator, we use the encoder and CLF with 1 hidden layer of 6 units each whereas for the cart pole system, we use the encoder and CLF with 2 hidden layers of 32 units each. For the decoder, we use a neural network with 2 hidden layers of 16 units each for all experiments.

Hyperparameter tuning: The hyperparameters α_1, α_2, α_3 and α_4 were tuned based on the combination of the controller performance and the empirical loss on the training data. We observed that for the Van der Pol system, $\alpha_1 = 0.001$, $\alpha_2 = 2$ $\alpha_3 = 1$ and $\alpha_4 = 1$ yielded the best results whereas for the cart pole we obtained the best results for $\alpha_1 = 0.05$, $\alpha_2 = 3$, $\alpha_3 = 1$ and $\alpha_4 = 1$. We use a trial-and-error method to guess the best hyperparameters. However, recent methods such as in [59] can be used to learn the optimal hyperparameters.

5. Conclusions

We proposed a Koopman operator based learning framework to compute a lifted bilinear system that serves as a higher dimensional representation of an unknown control-affine nonlinear system and design a stabilizing feedback controller based on a Control Lyapunov Function (CLF) which is computed under the same learning framework. Our approach simultaneously learns 1) the matrices that determine the state space model representation of the bilinear system, 2) the Koopman based observables and 3) a valid CLF by using a learner and a falsifier. The learned CLF is then used to design a stabilizing feedback controller (based on the learned Koopman bilinear system) which is then applied to the control-affine nonlinear system. Numerical experiments are provided to validate the ability of our proposed class of learning-based feedback controllers to stabilize control-affine nonlinear systems with unknown dynamics. A particularly exciting direction for our future work is to use the learned bilinear model to

design robust control laws for the unknown nonlinear system which can account for disturbances acted upon the latter system as well as modelling errors and uncertainties. Another possible direction is to extend the results presented herein to stochastic nonlinear control systems.

References

[1] K. K. Chen, J. H. Tu, C. W. Rowley, Variants of dynamic mode decomposition: boundary condition, Koopman, and fourier analyses, Journal of nonlinear science 22 (6) (2012) 887–915.

[2] J. H. Tu, Dynamic mode decomposition: Theory and applications, Ph.D. thesis, Princeton University (2013).

[3] M. Korda, I. Mezić, On convergence of extended dynamic mode decomposition to the Koopman operator, Journal of Nonlinear Science 28 (2) (2018) 687–710.

[4] M. O. Williams, M. S. Hemati, S. T. Dawson, I. G. Kevrekidis, C. W. Rowley, Extending data-driven Koopman analysis to actuated systems, IFAC-PapersOnLine 49 (18) (2016) 704–709.

[5] I. Abraham, T. D. Murphey, Active learning of dynamics for data-driven control using Koopman operators, IEEE Transactions on Robotics 35 (5) (2019) 1071–1083.

[6] D. Bruder, X. Fu, R. B. Gillespie, C. D. Remy, R. Vasudevan, Data-driven control of soft robots using Koopman operator theory, IEEE Transactions on Robotics 37 (3) (2020) 948–961.

[7] G. Mamakoukas, M. Castano, X. Tan, T. Murphey, Local Koopman operators for data-driven control of robotic systems, in: Robotics: science and systems, 2019.

[8] G. Mamakoukas, M. L. Castano, X. Tan, T. D. Murphey, Derivative-based Koopman operators for real-time control of robotic systems, IEEE Transactions on Robotics 37 (6) (2021) 2173–2192.

[9] V. Zinage, E. Bakolas, Koopman operator based modeling for quadrotor control on SE (3), IEEE Control Systems Letters 6 (2022) 752–757.

[10] M. Korda, Y. Susuki, I. Mezić, Power grid transient stabilization using Koopman model predictive control, IFAC-PapersOnLine 51 (28) (2018) 297–302.

[11] Y. Susuki, I. Mezić, Nonlinear Koopman modes and power system stability assessment without models, IEEE Transactions on Power Systems 29 (2) (2013) 899–907.

[12] A. Surana, M. O. Williams, M. Morari, A. Banaszuk, Koopman operator framework for constrained state estimation, in: 2017 IEEE 56th Annual Conference on Decision and Control (CDC), IEEE, 2017, pp. 94–101.

[13] M. Netto, L. Mili, A robust data-driven Koopman kalman filter for power systems dynamic state estimation, IEEE Transactions on Power Systems 33 (6) (2018) 7228–7237.

[14] H. Choi, U. Vaidya, Y. Chen, A convex data-driven approach for nonlinear control synthesis, Mathematics 9 (19) (2021) 2445.

[15] C. Folkestad, Y. Chen, A. D. Ames, J. W. Burdick, Data-driven safety-critical control: Synthesizing control barrier functions with Koopman operators, IEEE Control Systems Letters 5 (6) (2020) 2012–2017.

[16] D. Goswami, D. A. Paley, Bilinearization, reachability, and optimal control of control-affine nonlinear systems: A koopman spectral approach, IEEE Transactions on Automatic Control 67 (6) (2022) 2715–2728.

[17] B. Huang, X. Ma, U. Vaidya, Feedback stabilization using Koopman operator, in: 2018 IEEE Conference on Decision and Control (CDC), IEEE, 2018, pp. 6434–6439.

[18] S. Sinha, U. Vaidya, R. Rajaram, Operator theoretic framework for optimal placement of sensors and actuators for control of nonequilibrium dynamics, Journal of Mathematical Analysis and Applications 440 (2) (2016) 750–772.

[19] V. Zinage, E. Bakolas, Far-field minimum-fuel spacecraft rendezvous using Koopman operator and l 2/l 1 optimization, in: 2021 American Control Conference (ACC), IEEE, 2021, pp. 2992–2997.

[20] V. Zinage, E. Bakolas, Koopman operator based modeling and control of rigid body motion represented by dual quaternions, in: 2022 American Control Conference (ACC), IEEE, 2022, pp. 3997–4002.

[21] B. Lusch, J. N. Kutz, S. L. Brunton, Deep learning for universal linear embeddings of nonlinear dynamics, Nature communications 9 (1) (2018) 1–10.

[22] F. Fan, B. Yi, D. Rye, G. Shi, I. R. Manchester, Learning stable Koopman embeddings, arXiv preprint arXiv:2110.06509.

[23] M. Haseli, J. Cortés, Fast identification of Koopman-invariant subspaces: parallel symmetric subspace decomposition, in: 2020 American Control Conference (ACC), 2020, pp. 4545–4550.

[24] M. Haseli, J. Cortés, Learning koopman eigenfunctions and invariant subspaces from data: symmetric subspace decomposition, in: IEEE Transactions on Automatic Control, IEEE, 2021.

[25] M. Haseli, J. Cortés, Parallel learning of koopman eigenfunctions and invariant subspaces for accurate long-term prediction, IEEE Transactions on Control of Network Systems 8 (4) (2021) 1833–1845.

[26] M. Han, L. Zhang, J. Wang, W. Pan, Actor-critic reinforcement learning for control with stability guarantee, IEEE Robotics and Automation Letters 5 (4) (2020) 6217–6224.

[27] E. D. Sontag, A 'universal'construction of artstein's theorem on nonlinear stabilization, Systems & control letters 13 (2) (1989) 117–123.

[28] Z. Artstein, Stabilization with relaxed controls, Nonlinear Analysis: Theory, Methods & Applications 7 (11) (1983) 1163–1173.

[29] G. Chesi, D. Henrion, Guest editorial: Special issue on positive polynomials in control, IEEE Transactions on Automatic Control 54 (5) (2009) 935–936.

[30] D. Henrion, A. Garulli, Positive polynomials in control, Vol. 312, Springer Science & Business Media, 2005.

[31] Z. Jarvis-Wloszek, R. Feeley, W. Tan, K. Sun, A. Packard, Some controls applications of sum of squares programming, in: 42nd IEEE international conference on decision and control, Vol. 5, IEEE, 2003, pp. 4676–4681.

[32] A. Majumdar, R. Tedrake, Funnel libraries for real-time robust feedback motion planning, The International Journal of Robotics Research 36 (8) (2017) 947–982.

[33] A. A. Ahmadi, M. Krstic, P. A. Parrilo, A globally asymptotically stable polynomial vector field with no polynomial Lyapunov function, in: 2011 50th IEEE Conference on Decision and Control and European Control Conference, IEEE, 2011, pp. 7579–7580.

[34] A. Narasingam, J. S.-I. Kwon, Koopman Lyapunov-based model predictive control of nonlinear chemical process systems, AIChE Journal 65 (11) (2019) e16743.

[35] A. Narasingam, S. H. Son, J. S.-I. Kwon, Data-driven feedback stabilisation of nonlinear systems: Koopman-based model predictive control, International Journal of Control (2022) 1–12.

[36] S. H. Son, H.-K. Choi, J. Moon, J. S.-I. Kwon, Hybrid koopman model predictive control of nonlinear systems using multiple edmd models: An application to a batch pulp digester with feed fluctuation, Control Engineering Practice 118 (2022) 104956.

[37] G. Cybenko, Approximation by superpositions of a sigmoidal function, Mathematics of control, signals and systems 2 (4) (1989) 303–314.

[38] K. Hornik, Some new results on neural network approximation, Neural networks 6 (8) (1993) 1069–1072.

[39] C. Ya-Chien, R. Nima, G. Sicun, Neural Lyapunov control, in: NeurIPS, 2019.

[40] A. Mehrjou, M. Ghavamzadeh, B. Schölkopf, Neural Lyapunov redesign, in: Conference on Learning for Dynamics and Control, 2020.

[41] H. Ravanbakhsh, S. Sankaranarayanan, Learning control Lyapunov functions from counterexamples and demonstrations, Autonomous Robots 43 (2) (2019) 275–307.

[42] W. Jin, Z. Wang, Z. Yang, S. Mou, Neural certificates for safe control policies, arXiv preprint arXiv:2006.08465.

[43] A. Abate, D. Ahmed, M. Giacobbe, A. Peruffo, Formal synthesis of Lyapunov neural networks, IEEE Control Systems Letters 5 (3) (2020) 773–778.

[44] H. Dai, B. Landry, L. Yang, M. Pavone, R. Tedrake, Lyapunov-stable neural-network control, in: Robotics: Science and Systems, 2021.

[45] N. M. Boffi, S. Tu, N. Matni, J.-J. E. Slotine, V. Sindhwani, Learning stability certificates from data, in: Conference on Robot Learning, 2020.

[46] S. M. Richards, F. Berkenkamp, A. Krause, The Lyapunov neural network: Adaptive stability certification for safe learning of dynamical systems, in: Conference on Robot Learning, PMLR, 2018, pp. 466–476.

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

[48] D. Bruder, X. Fu, R. Vasudevan, Advantages of bilinear Koopman realizations for the modeling and control of systems with unknown dynamics, IEEE Robotics and Automation Letters 6 (3) (2021) 4369–4376.

[49] J. C. Butcher, The numerical analysis of ordinary differential equations: Runge-Kutta and general linear methods, Wiley-Interscience, 1987.

[50] S. L. Brunton, B. W. Brunton, J. L. Proctor, J. N. Kutz, Koopman invariant subspaces and finite linear representations of nonlinear dynamical systems for control, PloS one 11 (2) (2016) 150–171.

[51] E. D. Sontag, A Lyapunov-like characterization of asymptotic controllability, SIAM journal on control and optimization 21 (3) (1983) 462–471.

[52] S. Gao, S. Kong, E. M. Clarke, dReal: An SMT solver for nonlinear theories over the reals, in: International conference on automated deduction, Springer, 2013, pp. 208–214.

[53] L. d. Moura, N. Bjørner, Z3: An efficient smt solver, in: International conference on Tools and Algorithms for the Construction and Analysis of Systems, Springer, 2008, pp. 337–340.

[54] H. Barbosa, C. Barrett, M. Brain, G. Kremer, H. Lachnitt, M. Mann, A. Mohamed, M. Mohamed, A. Niemetz, A. Nötzli, et al., cvc5: a versatile and industrial-strength smt solver, in: International Conference on Tools and Algorithms for the Construction and Analysis of Systems, Springer, 2022, pp. 415–442.

[55] S. Gao, J. Avigad, E. M. Clarke, δ-complete decision procedures for satisfiability over the reals, in: International Joint Conference on Automated Reasoning, Springer, 2012, pp. 286–300.

[56] H. K. Khalil, Nonlinear systems, in: Pearson New York, 2015.

[57] D. Pylorof, E. Bakolas, Stabilization of input constrained nonlinear systems with imperfect state feedback using sum-of-squares programming, in: 2018

IEEE Conference on Decision and Control (CDC), IEEE, 2018, pp. 1847–1852.

[58] D. Pylorof, E. Bakolas, Safe nonlinear control design for input constrained polynomial systems using sum-of-squares programming, International Journal of Control 94 (9) (2021) 2603–2613.

[59] R. Bardenet, M. Brendel, B. Kégl, M. Sebag, Collaborative hyperparameter tuning, in: International conference on machine learning, PMLR, 2013, pp. 199–207.