Approximating Reachable Sets for Neural Network-Based Models in Real Time via Optimal Control

Omanshu Thapliyal, *Member, IEEE*, and Inseok Hwang, *Member, IEEE*

***Abstract*— In this brief, we present a data-driven framework for real-time estimation of reachable sets for control systems where the plant is modeled using neural networks (NNs). We utilize a running example of a quadrotor model that is learned using trajectory data via NNs. The NN learned offline can be excited online to obtain linear approximations for reachability analysis. We use a dynamic mode decomposition (DMD)-based approach to obtain linear lifting of the NN model. The linear models thus obtained can utilize optimal control theory to obtain polytopic approximations to the reachable sets in real time. The polytopic approximations can be tuned to arbitrary degrees of accuracy. The proposed framework can be extended to other nonlinear models that utilize NNs to estimate plant dynamics. We demonstrate the effectiveness of the proposed framework using an illustrative simulation of quadrotor dynamics.**

***Index Terms*— Approximation methods, machine learning, reachability analysis.**

I. Introduction

AS THE systems of interests of control engineers get more complex, and data get inexpensive to obtain in large quantities, machine learning finds increasing applications in control systems. Furthermore, obtaining simulated data for multiple trajectories of the system of interest is often easier than designing physical control. For instance, neural networks (NNs) often find applications to model plants, actuators, controller logic, and even for modeling the human operator's intent and logic. To this end, data-driven approaches to discover underlying physical models for dynamical systems are very useful.

Besides, set-based properties of safety, reachability, and controllability provide strong analytical bases to quantify system performance (especially under uncertainties). Of these, estimating reachability property is closely tied with other properties, such as viability, controllability, and safety [1]. In addition, reachability can be utilized for optimal control synthesis and high-level decision-making [2]. Reachable sets can be computed analytically by solving Hamilton Jacobi (HJ) partial differential equations (PDEs). Such HJ-based solutions of reachable sets are time-consuming and suffer from the "curse of dimensionality," making real-time applications difficult. To alleviate this, various numerical approximation techniques have been proposed to compute approximate reachable sets without explicit solutions to the associated HJ PDEs. These techniques employ polytopic approximations [3], Pontryagin's optimal control [4], and numerical differential equation solvers [5], to name a few. Based on these, a number of reachable set computation tools are available that utilize numerical techniques for approximate reachable sets and tubes (such as [6], [7], [8]), both forward and backward in time.

On the other hand, the ability to design control signals, and comment upon system properties under operational noise, parameter uncertainties, and system nonlinearities, is of a lot of importance. As a result, NNs are applied in the entire control system design process of system identification [9], output tracking [10], control synthesis [11], state estimation [12], and devising supervisory control logic [13]. Recently, NNs have been applied to learn nonlinear dynamical models of varying complexity. A robotic arm's inverse dynamics is inferred in [9] using iterative learning on real data from an iCub robotic arm. In [13], a learning-based scheme is used to infer supervisory control logic for cybersecurity analysis of supervisory control systems. Bansal et al. [14] synthesized control signals to control a quadrotor by learning its dynamics. More generally, utilizing machine learning techniques for system identification has been noted to be particularly useful in numerous recent system modeling and identification texts, such as [15], [16], and [17]. However, *despite the widespread usage of NNs* in solving dynamics and control problems, *the absence of reachable set computation/approximation tools for NN-based models prohibits a reliable application* of NN under uncertainties and operating conditions that demand safety guarantees. Since the machine learning models use data from an unknown dynamical system, numerical approaches to compute approximate reachable sets can be extended to the learned models themselves. To this end, the Koopman Operator theory has been used to learn NNs, as it provides a method to find a computationally scalable, equivalent linear lifted model [18]. Conversely, learning-based methods are also used to learn the Koopman operator itself, for control synthesis [19]. Nevertheless, linear control methods can be utilized on such linear lifted models to estimate reachable sets as polytopes, and an optimal control problem can be formulated to propagate these polytopes over time. The polytopic reachable set approximation can be made arbitrarily accurate [4]; therefore, the reliability of the proposed method is conditioned on the accuracy of the following: 1) the NN being able to approximate unknown plant models and 2) the Koopman operator being able to lift the NN model to a higher dimensional manifold and approximate it as a linear system.

Manuscript received 25 April 2022; revised 24 August 2022; accepted 26 December 2022. This work was supported in part by the NSF under Grant CNS-1836952. Recommended by Associate Editor Y. Pan. *(Corresponding author: Omanshu Thapliyal.)*

The authors are with the School of Aeronautics and Astronautics Engineering, Purdue University, West Lafayette, IN 47906 USA (e-mail: othapliy@purdue.edu; ihwang@purdue.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCST.2023.3234248.

Digital Object Identifier 10.1109/TCST.2023.3234248

A. Related Works

The general problem of computing output bounds of an NN can be readily related to reachable set computation for learned dynamical models. Output bound computation for NN-based controllers using specific activation functions has been achieved by solving mixed-integer linear programs (MILPs) [20] and relaxed linear programs (LPs) [21]. This makes the amenability toward real-time applications particularly low. Such methods exploit the properties of individual perceptrons in an NN, connected via rectified linear unit (ReLU) activations to obtain output bounds. Huang et al. [22] employ Bernstein polynomials to obtain Taylor approximations of NN-based control systems to obtain reachability flow pipes. In [23], exact reachable sets are computed for a control system employing ReLU activation functions-based NNs, with specific switched linear dynamics. The above methods utilize explicit system dynamics, or specific activation functions to obtain reachable sets for NN models. Most NNs employed to learn system dynamics can be arbitrarily nonlinear. On the other hand, reachable set computation/approximation using HJ methods or polytopes is extant in controls literature [1], [3], [4]. Being exact model-based methods, they rely on the complete knowledge of the dynamical modes of the system, an assumption no longer true for NN-based system modeling. To the best of our knowledge, there do not exist methods that extend optimal control theory based on local linear system approximations of the given NN to obtain approximate reachable sets for the NN models.

B. Contributions

The main contributions of the proposed method are listed as follows.

1) We utilize a dynamic mode decomposition (DMD)-based framework to obtain approximate linear models for the given learned nonlinear dynamics. This allows us to use optimal control theory to obtain polytopic approximations to the reachable sets for the approximately equivalent linear system.
2) The proposed framework is numerically efficient and much more amenable to real-time applications than solving MILPs or relaxed LPs at each time step. We demonstrate this using a realistic and detailed example of real-time reachable set approximation for a quadrotor model—a widely studied system for identification and control using NNs.
3) Finally, the framework can employ "plug-and-play" reachability modules from other reachability assessment tools for the approximately equivalent linear systems. That is, the introduced polytopic reachable set approximation methods can be replaced by other reachability modules, as shown in Fig. 1.

The rest of this brief is organized as follows. In Section II, we formulate the reachability problem for a learned model. Section III contains the main framework to estimate reachable sets for nonlinear models learned by an NN. The approximate reachable set computation is posed as an optimal control problem to obtain polytopic reachable sets. In Section IV, we implement the reachability estimation framework on an illustrative quadrotor example. We first consider a nominal quadrotor reachability case and then consider a separate scenario of estimating reachable sets when two of the rotors have failed. Finally, Section V presents our concluding remarks.

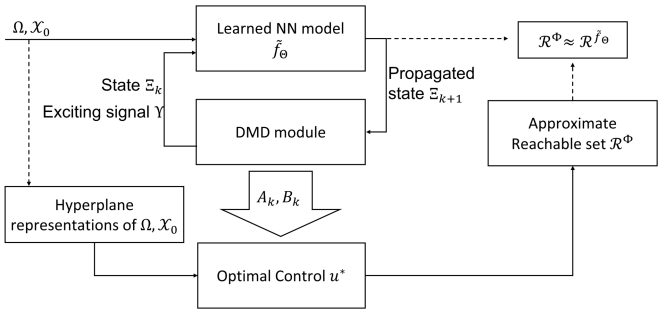

Fig. 1. Schematic of the proposed data-driven framework for approximate reachable set computation.

Notations: For two vectors u and v, their inner product is denoted by $\langle u, v\rangle$. For a matrix A, we denote its transpose by A^T and its Frobenius norm by $\|A\|_F$. For two sets A and B, we denote their Minkowski sum as $A \oplus B \triangleq \{a+b \,|\, a \in A, b \in B\}$. For a finite set A, if a random variable x is distributed uniformly in the set A, we write $x \approx \mathcal{U}_A$.

II. Problem Formulation

Consider the nonlinear dynamical system given as follows:

$$\dot{x}(t) = f(x(t), u(t)), \quad \text{and} \quad x(0) \in \mathcal{X}_0,\ u(t) \in \Omega,\ t \geq 0 \tag{1}$$

where $x \in \mathcal{X} \subseteq \mathbb{R}^{n_x}$ is the state, x_0 is the initial state in a known initial set $\mathcal{X}_0$, and the control input $u \in \mathbb{R}^{n_u}$ resides in the set Ω at all times t. Given some input-state data in the form of $X_k \triangleq \{x_0, \ldots, x_N\}$, $U_k \triangleq \{u_0, \ldots, u_N\}$ over multiple trajectories $k = 0, \ldots, n_T$, the unknown dynamical map f is learned using the trajectory data. The trajectory data is sampled from $\dot{x} = f(x, u)$ in (1) at some sampling rate Δt, such that $(x_i, u_i) \triangleq (x(i\,\Delta t), u(i\,\Delta t))$ for $i = 1, \ldots, N$.

A data-driven method, such as an NN, is employed to estimate the unknown dynamics f from the time series data trajectories as follows:

$$\dot{\tilde{x}}(t) = \tilde{f}_\Theta(\tilde{x}(t), u(t)). \tag{2}$$

Here, $\tilde{x}(t)$ is the state obtained by the NN from the data X_k, U_k, and it approximates the true state $x(t)$ as long as $\tilde{f}_\Theta \approx f$. The approximate dynamics $\tilde{f}_\Theta$ is parameterized by Θ, which contains the parameters of the learning method employed (i.e., NN weights and biases). Without loss of generality, we assume the initial state $\tilde{x}(0) \in \mathcal{X}_0$ and control input $u \in \Omega$. The reachability problem for the NN model is then to find

$$\mathcal{R}^{\tilde{f}}(\tau; \mathcal{X}_0) \triangleq \left\{\tilde{x}(\tau) \,|\, \dot{\tilde{x}} = \tilde{f}_\Theta(\tilde{x}, u), \tilde{x}(0) \in \mathcal{X}_0, u \in \Omega; \Theta\right\} \tag{3}$$

at some time τ, given initial conditions $\mathcal{X}_0$ and admissible control set Ω. Note that the problem to compute $\mathcal{R}^{\tilde{f}}(\tau; \mathcal{X}_0)$

is nontrivial due to the arbitrary, unstructured nonlinearities present in the NN modeling (parameterized by Θ).

Remark 1 (NNs as Universal Approximator [24]): Given time series trajectory data X_k, U_k, a causal NN with parameters Θ can be employed to approximate f appropriately. Obviously, $\tilde{f}_\Theta \to f$ as the number of available trajectories $n_T \to \infty$; therefore, $\tilde{x}(\tau) \to x(\tau)$ for $0 \le \tau \le N\Delta t$.

So, if we can find the set $\mathcal{R}^{\tilde{f}}(\tau; \mathcal{X}_0)$, it gives us a good approximation of the reachable set for the nonlinear system in (1). Since we do not concern ourselves with devising a new learning scheme, we shall restrict our discussion to $\mathcal{R}^{\tilde{f}}(\tau; \mathcal{X}_0)$. Moving forward, *we assume the dynamics of the system to be given by $\tilde{f}_\Theta$, as the NN can be trained accurately due to Remark 1.*

III. Reachability Framework for NN Models

In this section, we will assume the nonlinear dynamical system has been learned using available NN techniques and focus on approximating reachable sets of NN dynamics $\tilde{f}_\Theta$ using a relatively computationally cheap method. To this end, we revise a formulation of DMD that allows for control inputs [25]. This allows us to build finite-dimensional approximation to the infinite-dimensional Koopman operator, to get equivalent linear time-varying system models.

A. DMD With Control

Let us look at a data-driven method for approximating the Koopman Operator, called DMD. Nominal forms of DMD involve trajectory data (called "snapshots") consisting of state evolutions over time x_k. The trajectory data get mapped under a linear operator as $x_{k+1} \approx Ax_k$. DMD with control (DMDc) was proposed to include input-state relations to such trajectory evolutions in [25]. Given an input-state data point x_k, u_k, DMDc attempts to find the pair of operators A, B, such that $x_{k+1} \approx Ax_k + Bu_k$ for data points on state $x_k \in \mathbb{R}^{n_x}$ and input $u_k \in \mathbb{R}^{n_u}$. The data matrices at time step k are temporal snapshots of the trajectory, of width $w \in \mathbb{N}$, given by

$$\Xi_{k,w} \triangleq \begin{bmatrix} | & & | \\ x_k & \cdots & x_{k+w} \\ | & & | \end{bmatrix}, \quad \Upsilon_{k,w} \triangleq \begin{bmatrix} | & & | \\ u_k & \cdots & u_{k+w} \\ | & & | \end{bmatrix}. \tag{4}$$

The snapshot with data points propagated one step in time can then be represented as follows:

$$\Xi_{k+1,w} = \Gamma_{k,w} \begin{bmatrix} \Xi_{k,w} \\ \Upsilon_{k,w} \end{bmatrix}, \quad \text{where } \Gamma_{k,w} \triangleq \begin{bmatrix} A & B \end{bmatrix}. \tag{5}$$

Note that the mapping Γ varies over time and is parameterized by the snapshot width w. The DMDc solution to (5) can be viewed as a least-square regression problem to find a $\Gamma \in \mathbb{R}^{n_x \times (n_x + n_u)}$, such that

$$\Gamma_{k,w} = \arg\min \left\| \Xi_{k+1,w} - \Gamma_{k,w} \begin{bmatrix} \Xi_{k,w} \\ \Upsilon_{k,w} \end{bmatrix} \right\| F, \quad \text{or}$$
$$\Gamma_{k,w} = \begin{bmatrix} A & B \end{bmatrix} = \Xi_{k+1,w} \begin{bmatrix} \Xi_{k,w} \\ \Upsilon_{k,w} \end{bmatrix}^\dagger \tag{6}$$

where $[\cdot]^\dagger$ denotes the pseudoinverse. Extracting columns from the least-square solution in (6), we get a linear time-varying system, such that $x_{k+1} \approx A_k x_k + B_k u_k$.

A numerically efficient way to compute the least-square solution utilizes singular-value decompositions of data snapshot matrices [25]. Over time, matrices A_k, B_k can be estimated in real time with a time-moving window of width w. In our case, the moving window obtains snapshot data from the input-state relation learned by the NN. Note that the state snapshot data can be easily obtained by exciting the learned model $\tilde{f}_\Theta$ by sampling an arbitrary control input from Ω that forms $\Gamma_{k,w}$ and noting NN output into the propagated state data snapshot $\Xi_{k+1,w}$. This gives an independent framework that uses excitations of the NN model to obtain approximate linear models, depicted in Fig. 1.

B. Approximate Reachable Set Computation Using DMDc Model

Now that we have linear approximations of the form $x_{k+1} = A_k x_k + B_k u_k$, and we will focus on obtaining reachable sets for the linear time-varying system $(A(t), B(t))$. Here, the system matrices $A(t)$ and $B(t)$ satisfy $\exp\{A(t)\Delta t\} = A_k$ and $\int_0^{\Delta t} \exp\{A(s)s\} B(s) \mathrm{d}s = B_k$ for $t \in [k\Delta t, (k+1)\Delta t)$. That is, A_k and B_k get updated at time step $k+1$ upon receiving new snapshot data via (5); hence, $A(t)$ and $B(t)$ get updated at time $(k+1)\Delta t$. Let $\Phi(t, 0)$ be the state-transition function associated with the linear system $(A(t), B(t))$. That is, $\dot{\Phi}(t, 0) = -A(t)\Phi(t)$, and $\Phi(0, 0) = I$. Let $\mathcal{R}^\Phi(\tau; \mathcal{X}_0, \Omega)$ be the reachable set of the system $(A(t), B(t))$ at time τ. If the DMDc approximation is accurate, it should suffice to concern ourselves with approximating $\mathcal{R}^\Phi(\tau; \mathcal{X}_0, \Omega)$.

Without loss of generality, we assume that the admissible control set and the initial state set are polytopes as follows:

$$\mathcal{X}_0 = \bigcap_{i=1}^{n_1} \left\{ v \in \mathbb{R}^{n_x} \mid \langle c_i(0), v \rangle \le \gamma_i(0) \right\}$$
$$\Omega = \bigcap_{i=1}^{n_2} \left\{ u \in \mathbb{R}^{n_u} \mid \langle d_i, u \rangle \le \varepsilon_i \right\} \tag{7}$$

defined for arbitrary vectors v. In addition, c_i and d_i are normal vectors parameterizing the hyperplanes defining each face of the polytopes in (7). Let the hyperplanes $H_i \equiv \langle c_i(0), v \rangle = \gamma_i(0)$ touch the set $\mathcal{X}_0$ at points $x_i^*(0)$ for $i = 1, \ldots, n_1$. The above polytopic assumption is not limiting, as arbitrarily convex, compact sets $\mathcal{X}_0$ and Ω can be bounded by tight polytopes as (7). Clearly, the following hold true:

$$x_i^*(0) = \arg\max_{v \in \mathcal{X}_0} \{\langle c_i(0), v \rangle\}, \text{ and}$$
$$\gamma_i(0) = \max_{v \in \mathcal{X}_0} \{\langle c_i(0), v \rangle\}. \tag{8}$$

A polytopic reachable set approximation looks at only the points of contact $x_i^*(0)$ of reachable sets of the linear system (see [3], [4]). Similarly, let $x_i^*(\tau)$ be the point of contact of $\mathcal{R}^\Phi(\tau; \mathcal{X}_0, \Omega)$ at the hyperplane $H_i(\tau)$ defined as follows:

$$H_i^*(\tau) = \left\{ x \mid \langle c_i(\tau), x_i^*(\tau) \rangle = \gamma_i(\tau) \right\} \tag{9}$$

for $i = 1, \ldots, n_1$, at time $\tau > 0$. Such an argument can be made for compact and convex sets $\mathcal{X}_0$ and Ω. From [4],

this ensures that the reachable set $\mathcal{R}^{\Phi}$ remains compact and convex at all times. This allows us to find hyperplanes $H_i^*(\tau)$ to support the reachable set. Using an argument similar to (8), the point of contact between $H_i^*(\tau)$ and $\mathcal{R}^{\Phi}$ at an arbitrary time τ can be defined as follows:

$$\begin{aligned} x_i^*(0) &= \underset{v\in\mathcal{R}^{\Phi}(\tau;\mathcal{X}_0,\Omega)}{\arg\max} \langle c_i(\tau), v\rangle \\ &= \arg\max\Bigg\{\langle c_i(\tau), v\rangle \ \text{ s.t. } v(t)=\Phi(t)x(0) \\ &\qquad + \int_0^t \Phi(t,s)B(s)u(s)ds, u(t)\in\Omega, t\le\tau\Bigg\}. \end{aligned} \tag{10}$$

Also, from (8), the distance between $H_i^*(\tau)$ and $\mathcal{R}^{\Phi}$ can be expressed as follows:

$$\begin{aligned} \gamma_i(\tau) &= \max_{v\in\mathcal{R}^{\Phi}(\tau;\mathcal{X}_0,\Omega)} \langle c_i(\tau), v\rangle \\ &= \max\Bigg\{\langle c_i(\tau), v\rangle \ \text{ s.t. } v(t)=\Phi(t)x(0) \\ &\qquad + \int_0^t \Phi(t,s)B(s)u(s)ds, u(t)\in\Omega, t\le\tau\Bigg\}. \end{aligned} \tag{11}$$

For a given set of initial points of contact $x_i^*(0)$, (10) and (11) depend only on the choice of $u \in \Omega$, thereby forming an optimal control problem.

Theorem 1: Let the optimal control $u_i^*(\tau)$ be the solution to $\arg\max_{u(\tau)} \langle c_i(\tau), B(\tau)u(\tau)\rangle$. Then, for $\dot{c_i} = -A(\tau)^T c_i(\tau)$ with the initial condition $\gamma_i^*(0)$, the hyperplane $H_i^*(\tau) \equiv \langle c_i(\tau), x^*(\tau)\rangle$ supports reachable set $\mathcal{R}^{\Phi}$.

Proof: The proof follows from Pontryagin's maximum principle [3]. The contact point $x_i^*(\tau)$ evolves as $\dot{x_i^*} = A(t)x(\tau) + B(t)u_i^*(\tau)$ for the given optimal control. For the linear system $A(t), B(t)$, the costate $\lambda(t)$ evolves as $\dot{\lambda}(t) = A(t)^T\lambda(t)$. Choose $c_i(\tau) = \lambda_i(\tau)$, where $\dot{\lambda_i}(\tau) = -A(\tau)^T\lambda_i(\tau)$ combined with the initial condition $\lambda_i(0) = \gamma_i^*(0)$ and suppress time indices for brevity. Note the time derivative of $\langle\lambda_i, x\rangle$ equals

$$\begin{aligned} \frac{d}{d\tau}\langle\lambda_i, x\rangle &= \langle\dot{\lambda_i}, x\rangle + \langle\lambda_i, \dot{x}\rangle \\ &= \langle -A^T\lambda_i, x\rangle + \langle\lambda_i, Ax + Bu\rangle = \langle\lambda_i, u\rangle \le \langle\lambda_i, u_i^*\rangle \\ &\Rightarrow \frac{d}{d\tau}\langle\lambda_i, x_i^*\rangle = \frac{d}{d\tau}\gamma^*. \end{aligned}$$

Combined with the initial conditions on the points of contact, i.e., $\langle\lambda_i(0), x_i^*(0)\rangle = \gamma_i^*(0)$, one gets $\langle\lambda_i(\tau), x_i(\tau)\rangle \le \langle\lambda_i(\tau), x_i^*(\tau)\rangle = \gamma_i^*(\tau)$. Hence, the hyperplane defined by c_i^* and x_i^* touches the reachable set. ∎

Remark 2: From [4], the polytopic approximation can be made arbitrarily accurate. In fact, at time τ

$$\text{convex hull}\left\{x_1^*, \ldots, x_{n_1}^*\right\} \subset \mathcal{R}^{\Phi}(\tau) \subset \cap_1^{n_1}\{\lambda_i, x\} \le \gamma_i^*$$

that is, the convex hull of the supporting points provides an under-approximation of the reachable set. At the same time, the hyperplanes provide the over-approximation of the same.

The polytopic reachable set approximation is a well-known numerically efficient method that is utilized by numerous reachable set computation tools. The τ-time reachable "tube" resulting from this method can be defined as the Minkowski sum $\oplus_{s=0}^{\tau}\mathcal{R}^{\Phi}(s;\mathcal{X}_0,\Omega)$. Combined with the DMDc-based method to obtain linear approximations of the NN model, this provides us with a scalable way to estimate reachable sets for NN models. Clearly, the proposed numerical method relies on the universal approximation capabilities of NNs [24]. Similarly, DMDc (more generally, DMD) converges in operator norm to the Koopman operator with increasing number of data points [26]. Indeed, given enough data points, and snapshot widths big enough, arbitrarily accurate reachable set approximations can be achieved. This is true, in general, for most data-driven schemes. In reality, the proposed framework provides a computationally cheap way to compute approximate reachable sets for the learned models, as the only additional computation steps involve matrix inversions in DMDc, and matrix exponentiation in propagating λ.

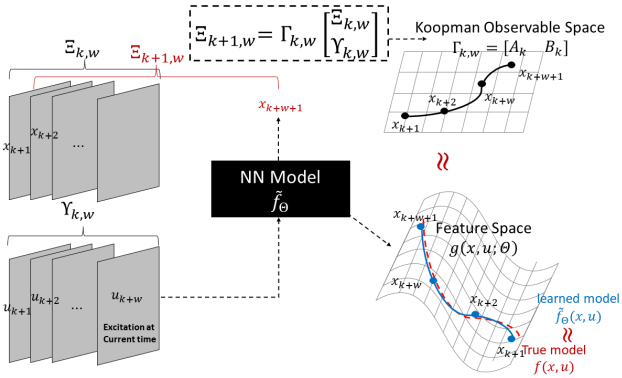

Fig. 2. Comparison of the temporal data snapshots, as learned by the NN versus the DMDc approximation.

A depiction of the data snapshots and the subsequent mappings discovered by the NN model and the DMDc method are shown in Fig. 2. The learned model attempts to take temporal trajectory data to a higher dimensional manifold, usually governed by the so-called "feature space," parameterized by the NN parameters Θ. On the other hand, the DMD-based method attempts to find approximations to the infinite-dimensional Koopman operator, which considers the temporal trajectories in some "observable space," where the trajectory evolution is (approximately) an action of a linear operator [25], [27]. This is because DMD tries to find finite-dimensional truncations of the Koopman operator. Hence, the proposed framework is amenable to a real-time implementation and is presented in Section IV for a quadrotor.

Remark 3: Once the model is learned, the computational expense is on the order of $\mathcal{O}([\cdot]^{-1}) + \mathcal{O}(\exp[\cdot]) \approx \mathcal{O}(\exp[\cdot])$. These can be accomplished in relatively computationally inexpensive ways by existing linear algebra libraries (such as PyLops [28] and Armadillo [29]).

IV. Reachable Sets for a Quadrotor

Although locally linear models of the quadrotor are often used for control synthesis [30], a fully nonlinear quadrotor model is required to capture its wide dynamical range. We demonstrate our proposed framework using a fully

nonlinear, 12 degree-of-freedom (DOF) quadrotor model, based on [31]. In this section, we will look at the reachable set computation problem of the 12 DOF, nonlinear dynamics, for a given $\mathcal{X}_0$ and Ω over time.

The state vector $\xi \in \mathbb{R}^{12}$ is given by the 3-D position $[x, y, z]^T$ and its respective velocities $[\dot{x}, \dot{y}, \dot{z}]^T$ and the 3-D angular attitude $[\phi, \psi, \theta]^T$, and respective angular velocities $[\dot{\phi}, \dot{\psi}, \dot{\theta}]^T$. The 12 DOF state ξ evolves as follows:

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} -\frac{u_1}{m}(\sin\phi\cos\psi + \cos\phi\cos\psi\sin\theta) \\ -\frac{u_1}{m}(\cos\phi\sin\psi\sin\theta - \cos\psi\sin\psi) \\ g - \frac{u_1}{m}(\cos\phi\cos\theta) \end{bmatrix} \quad (12)$$

$$\begin{bmatrix} I_{xx}\ddot{\phi} \\ I_{yy}\ddot{\theta} \\ I_{zz}\ddot{\psi} \end{bmatrix} = \begin{bmatrix} u_2 \\ u_3 \\ u_4 \end{bmatrix} - \begin{bmatrix} (I_{zz} - I_{yy})\dot{\theta}\dot{\psi} \\ (I_{xx} - I_{zz})\dot{\phi}\dot{\psi} \\ (I_{yy} - I_{xx})\dot{\theta}\dot{\phi} \end{bmatrix} \quad (13)$$

where I_{xx}, I_{yy}, and I_{zz} are the moments of inertia along the three axes, g is the acceleration due to gravity, and m is the quadrotor's mass. Variables $u_1, \ldots, u_4$ relate to the actual angular velocity command at the four rotors $\omega_1, \ldots, \omega_4$ as follows:

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} k_f & k_f & k_f & k_f \\ -lk_f & lk_f & lk_f & -lk_f \\ lk_f & lk_f & -lk_f & -lk_f \\ k_m & -k_m & k_m & -k_m \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix} \quad (14)$$

via the aerodynamic force and moment constants k_f and k_m, respectively, and the distance of rotors from the center l (see [31] for the derivation details). Determining the force and moment constants k_f and k_m, in itself, requires experimental system identification, let alone the nonlinear dynamics in (12)–(14). As a result, learning the nonlinear dynamics for a quadrotor has been of interest in multiple works for control synthesis.

We assume that we have access to n_T number of input-state trajectories X_k, U_k, each starting with a randomly initialized state $\xi_0 \approx \mathcal{U}_{\mathcal{X}_0}$. We consider an NN capable of learning approximate dynamics $\tilde{f}_\Theta$ from the time series input-state trajectories X_k, U_k. To this end, we employ a causal multistep NN to recover the nonlinear dynamics $\tilde{f}_\Theta$, based on [32]. The key idea of a multistep NN is to use time series trajectory over a number of steps, say $\xi_k, \xi_{k-1}, \ldots, \xi_{k-m}$ and $\omega_{i,k}, \ldots, \omega_{i,k-m}$ for $i = 1, \ldots, 4$, and find appropriate weights for a nonlinear function that takes the m-step trajectory and maps it to ξ_{k+1} (see [32] for more details). The actual functional approximation is offloaded to the multistep NN whose weights Θ minimize the mean-squared error over each m-step slice of the trajectory data, for a fixed m.

Here, we employ a multistep NN, which is a multilayer perceptron NN based on [32], that is used to perform system identification for differential equation-based dynamical systems. The NN attempts to approximate the system dynamics $\dot{x} = f(x)$ from its trajectory data $\{x_k\}_{k=0}^N$ by "unrolling" a trajectory of length N and approximating the dynamical map $\tilde{f}_\Theta : x_k \to x_{k+1}$ for $0 \le k \le N-1$. We used a multistep NN with three layers, including one hidden layer (12, 256, and 12 neurons, respectively). The multistep scheme used was

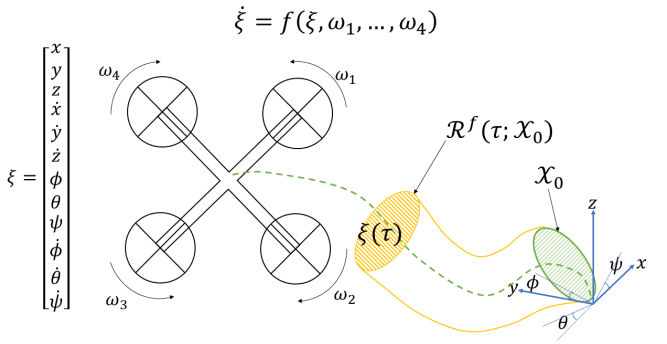

Fig. 3. Reachable set computation problem for a quadrotor: state ξ at time τ along some trajectory, starting from some initial set $\mathcal{X}_0$.

Adams–Moulton (i.e., using the trapezoidal rule to extend the function between k and $k+1$) to train over 100 trajectories, with discretization time $\Delta T = 0.1$s, activation function $\tanh(\cdot)$, and mean square error (MSE) loss function with adaptive moment estimation (ADAM) as the optimizer. The multistep NN took ≈80 s to train, using Python, on an Intel Xeon CPU running at 2.20 GHz with a 13-GB memory and a 56-MB cache size and was trained over 2000 epochs and converges to an MSE loss of 1.26×10^{-3}.

A. Reachable Set Computation

An example of the reachable set computation problem for the quadrotor model in (12)–(14) is depicted in Fig. 3. A randomly chosen initial state starts from a given initial set (shown in green), and the collection of all possible evolutions of the quadrotor's state after time τ resides in the set $\mathcal{R}^{\tilde{f}}(\tau; \mathcal{X}_0, \Omega, \Theta)$ (shown in yellow). In addition, the command input to each rotor has an additive noise as follows:

$$\omega_i(t) = v_i(t) + w_i, w_i(t) \sim \mathcal{U}_{[-0.25,0.25]} \text{ for } i = 1, \ldots, 4 \quad (15)$$

and the admissible control set is defined as follows:

$$\Omega \triangleq \{\omega_1, \ldots, \omega_4 \,|\, \omega_i(t) = v_i(t) + w_i, \forall i\} \quad (16)$$

where the additive noise w_i is assumed to be independent, identically distributed at all times t. Equations (15) and (16) allow us to model the actuator noise into the reachable set approximation problem. That is, $\mathcal{R}^{\tilde{f}}(\tau; \mathcal{X}_0, \Omega, \Theta)$ contains all possible states that can be reached in time τ, starting with $\xi_o \in \mathcal{X}_0$, under the noisy rotor command input set Ω.

We first learn the nonlinear model $\tilde{f}_\Theta$ for $n_T = 100$ trajectories, each initialized with a random position $x, y, z \approx \mathcal{U}_{[-0.5,0.5]}$ coordinate (in m) and a random pose $\phi, \psi, \theta \approx \mathcal{U}_{[-0.1,0.1]}$ (in radians). This defines the initial set $\mathcal{X}_0$ and also provides explicit forms of hyperplanes H_i at time $\tau = 0$. To generate the training and testing datasets, we generated trajectories, sampled randomly to initiate from $\mathcal{X}_0$, and applied control sequences sampled randomly from Ω (as shown in Fig. 5) to each of the four rotors.

Based on the DMDc-based framework in Section III, approximate linear time-varying models (A_k, B_k) are developed for $k = i\Delta t$ and a sampling time of $\Delta t = 0.1$s and $k = 0, \ldots, 50$. A comparison of the true, unknown position

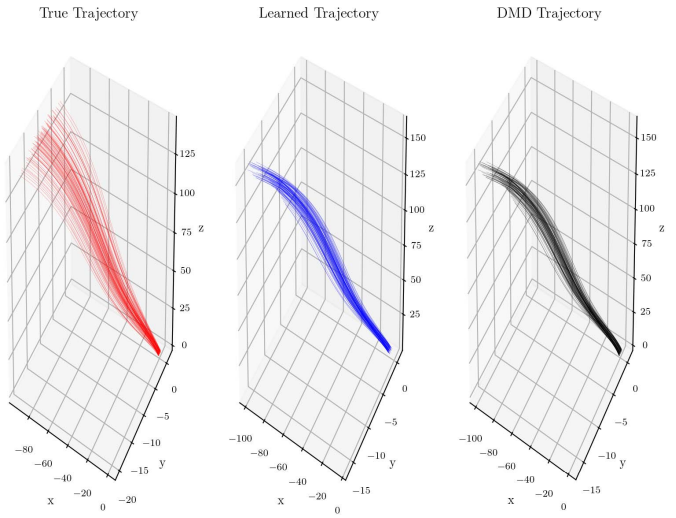

Fig. 4. 3-D state trajectories: true trajectories (in red), trajectories reconstructed by the multistep NN (in blue), and DMDc reconstructed trajectories (in black).

trajectories under mapping f, the predicted trajectories under learned model $\tilde{f}_\Theta$, and the trajectories reconstructed using DMDc under the mapping A_k, B_k are shown in Fig. 4. Clearly, given a relatively small data set $n_T = 100$, the multistep NN is able to reconstruct the true trajectories accurately. In addition, despite the nonlinearities in the learned model $\tilde{f}_\Theta$, for the given small DMDc width $n_w = 8$, DMD reconstructions are also observed to be accurate.

Next, we represent the initial set $\mathcal{X}_0$ using $n_1 = 24$ hyperplanes and the admissible control set Ω using $n_2 = 8$ hyperplanes. Each hyperplane supports $\mathcal{X}_0$ at a contact point ξ_i^*, and the τ-time reachability problem becomes the τ-time optimal control problem using the lifted system A_k, B_k. That is, at a time $t = k\Delta t$, the lifted model A_k, B_k is used to solve the optimal control problem in (10) and (11), propagating $\xi_i^*(k\Delta t)$ under the optimal control input $\omega_i^*(t) = \omega_i^*$ for $k\Delta t \le t < k\Delta t + \tau$. Propagating each contact point for time t to $t+\tau$ gives the supporting structure for the τ time reachable set. Given enough data, the multistep NN model approaches the true, unknown dynamical map, and the DMD lifted system approaches the learned model. This accuracy in approximation is validated in Figs. 5 and 6.

The admissible control set Ω is given by $v_i(t)$ defined as the sinusoidal angular velocity inputs to the four rotors with an added noise, as shown in Figs. 5 and 6 (green envelope). The solid green plots to the right in the figures depict the optimal control input that propagates an arbitrary hyperplane's contact point ξ_i^* over time, by applying rotor control $\omega_1^*, \ldots, \omega_4^*$, as shown. The points of contact of the hyperplanes are depicted in red, at each time step $0, \Delta t, 2\Delta t, \ldots$ for a simulation duration of 5 s. Inner approximations to the reachable tubes are simply Minkowski sums of the convex hulls in red. An arbitrary trajectory starting from an $\xi(0)$ sampled from $\mathcal{U}_{\mathcal{X}_0}$ is reconstructed using the multistep NN, shown as a black solid line.

Despite the reasonable accuracy of the trained multistep NN (in Fig. 4), DMDc reconstruction finds accurate equivalent

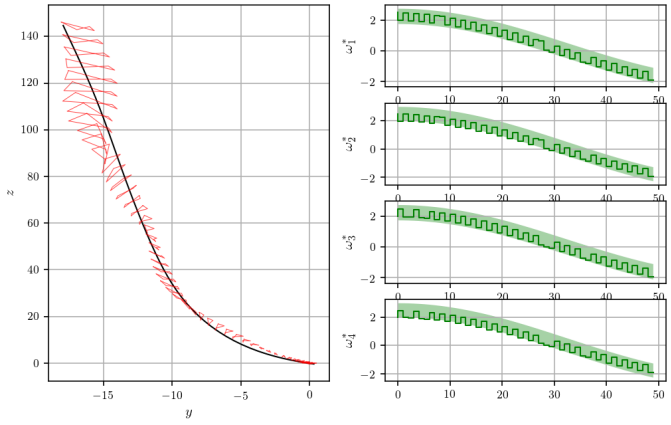

Fig. 5. Inner approximations of reachable sets (convex hulls of points ξ_i^*) in the y–z plane.

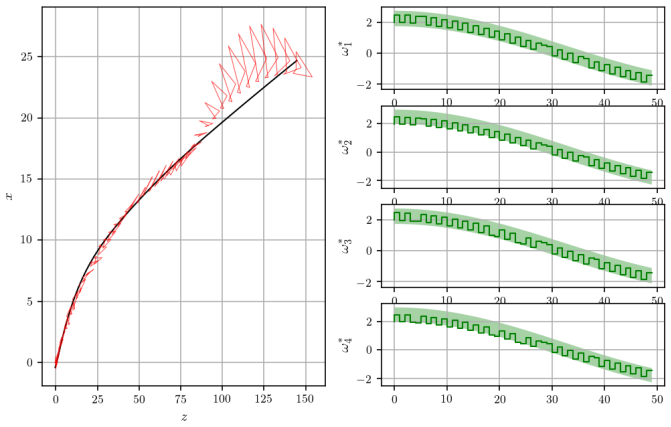

Fig. 6. Inner approximations of reachable sets (convex hulls of points ξ_i^*) in the z–x plane.

TABLE I

COMPARING LP, MILP, AND THE PROPOSED METHODS

Method	NN model	most expensive operation	Computational Cost w.r.t. #layers	Computational Cost w.r.t. #variables
LP method [21]	required	solve LP	$\mathcal{O}(L)$	$\mathcal{O}(n^\alpha)$
MILP method [20]	required	solve MILP	$\mathcal{O}(L)$	exponential
Proposed method	X	$\exp[\cdot]$	$\mathcal{O}(1)$	$\mathcal{O}(n^3)$

linear system models and is, therefore, able to find approximate reachable sets that are fairly accurate, in real time. As noted in Section I-A, there do not exist related methods that extend optimal control theory to obtain approximate reachable sets for NN models. The closest methods require exact, detailed NN models while employing LPs [21] and MILPs [20] to obtain reachable sets or output bounds. As the proposed method does not require any internal details of NN architecture, treating it as a black box, a direct comparison is not very meaningful. However, one can compare computational costs of the said methods as noted in Table I.

Note that each layer L in the NN architecture with n variables introduces $\mathcal{O}(Ln)$ variables for the LP- and MILP-based methods. Since the MILP problems are NP-hard, the method in [20] has a worst case complexity as bad as brute force search (hence, an exponential worst case computational complexity). The LP-based method has a computational complexity of $\mathcal{O}[L(4nL)^{2.5}]$ (i.e., the computational cost of solving

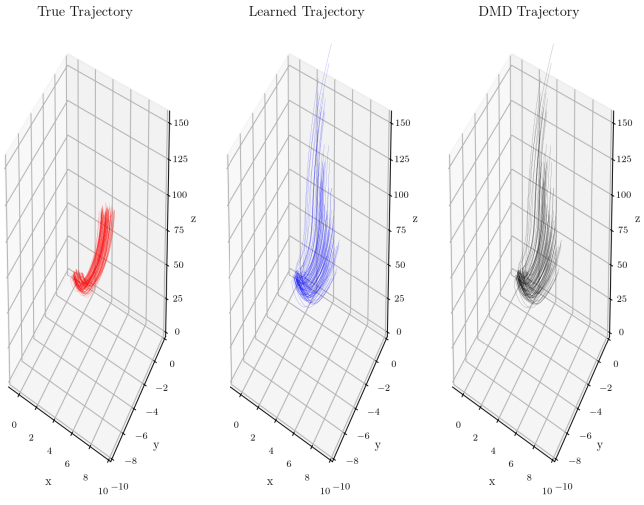

Fig. 7. 3-D state trajectories under rotor failure at rotors 2 and 3: true trajectories (in red), trajectories reconstructed by the multistep NN (in blue), and DMDc reconstructed trajectories (in black).

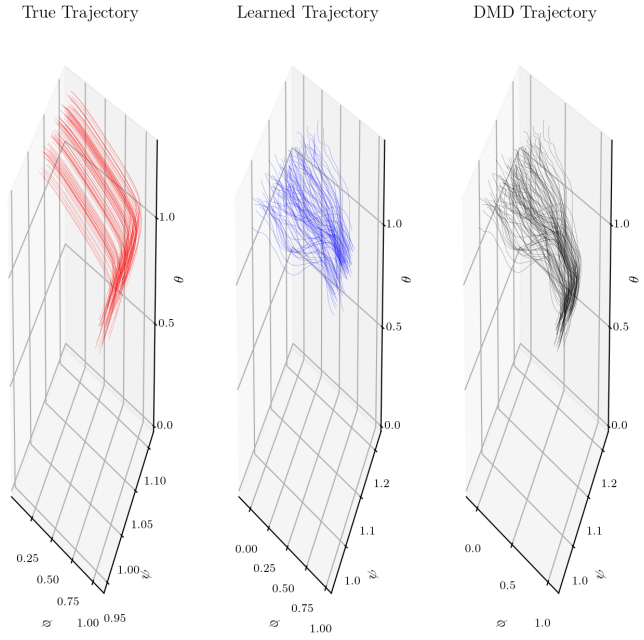

Fig. 8. 3-D attitude trajectories under rotor failure at rotors 2 and 3: true trajectories (in red), trajectories reconstructed by the multistep NN (in blue), and DMDc reconstructed trajectories (in black).

one LP for each layer). On the other hand, the associated computational costs for the proposed method are based only on the cost of $\exp[\cdot] \approx \mathcal{O}[(n+n_h)^3]$ for n_h hyperplanes. Despite being computationally cheaper, the LP-based method provides hyperrectangular approximations to the reachable sets (hence, loose overapproximations) and is closer to interval reachable set methods, such as [33] and [34]. Also, introducing the MILP and LP encodings requires extra overhead computations. Finally, the propagation of the points of contact for each hyperplane can be done independently. Therefore, the proposed method is more amenable to parallelization, as opposed to the interlayer dependency of variables in the MILP and LP formulations.

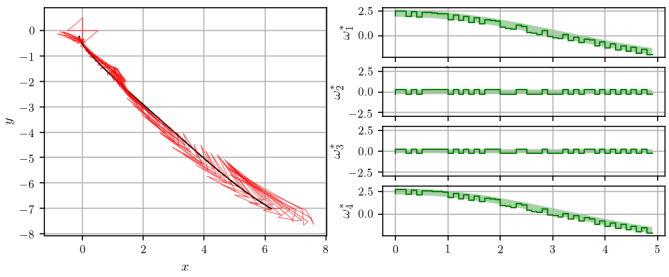

Fig. 9. Rotor failure at rotors 2 and 3: inner approximations of reachable sets (convex hulls of points ξ_i^*) in the x–y plane.

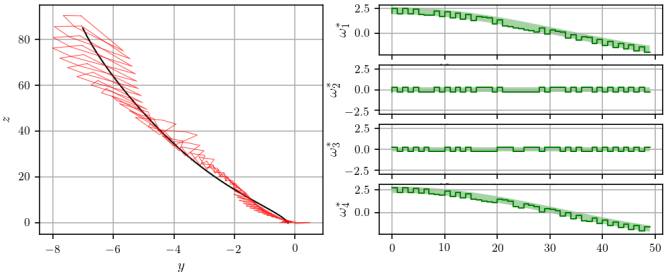

Fig. 10. Rotor failure at rotors 2 and 3: inner approximations of reachable sets (convex hulls of points ξ_i^*) in the y–z plane.

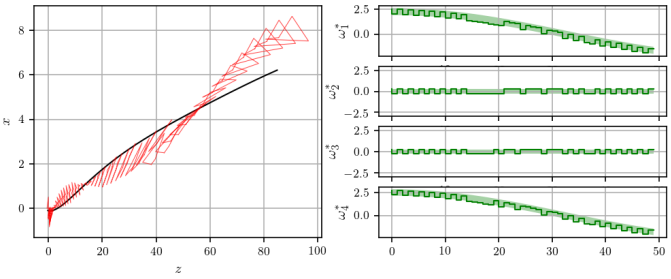

Fig. 11. Rotor failure at rotors 2 and 3: inner approximations of reachable sets (convex hulls of points ξ_i^*) in the z–x plane.

B. Reachable Set Computation Under Rotor Failure

Computing reachable sets under actuator failures is an important step to assess the compromised system's capabilities. In this scenario, rotors 2 and 3 suffer from a total failure and appear only as noise in the input channel. This significantly modifies the admissible control set $\mathbf{\Omega}$. Due to the rotor failures, the NN model strays from the true trajectory, as shown in Fig. 7. This is further exaggerated in the ϕ, ψ, and θ trajectories, shown in Fig. 8. Note that both the NN model and the DMDc model are not close to the true angular pose trajectories, but the DMDc model is still close to the learned model.

As a result of the rotor failure, the quadrotor's reachable sets are very different from the nominal case. The admissible control sets $\mathbf{\Omega}$ [shown in Figs. 9–11 (green envelopes)] are completely different due to ω_2 and ω_3 being bounded noise, while the remaining rotors provide the nominal angular velocity command. This is also reflected in the resultant optimal control for an arbitrary hyperplane's point of contact, shown in the solid green lines. The convex hulls of the points of contact denote inner approximations of the reachable sets.

Despite the highly nonlinear model, including rotor failures, and relatively small data set (both n_T and n_w), reachable set

computation can be achieved with relatively high accuracy and at a per-step computational increment of <0.5 s Therefore, the proposed method works well for real-time approximations of reachable sets.

V. Conclusion

In this brief, we presented a novel, data-driven framework for computing approximate reachable sets for nonlinear models learned using neural networks. A computationally efficient lifting-based method was proposed to find linear approximations of the learned model by exciting the NN at each time step. The proposed data-driven scheme was demonstrated to be computationally cheap and was found to be amenable to usage in conjunction with other reachability tool sets. Being data-driven, the proposed framework can be made more accurate if more data are available. Moreover, real-time application of the framework was observed in a realistic quadrotor example. Approximate reachable sets were computed for a causal NN trained to learn the quadrotor's dynamics. In addition, modified reachable sets were computed for the quadrotor in a scenario that models rotor failures. The proposed scheme was demonstrated to be of particular importance to safety critical scenarios where real-time approximation and update of reachable sets are required.

As our immediate future work, we plan to investigate the space complexity of the framework and comment upon data required for a predescribed level of accuracy in reachable set computation. Similarly, we plan to examine space complexity for a desired level of robustness against parameter variation. We also plan to look into developing efficient codes to facilitate a wrapper allowing plug-and-play usage with existing reachability toolboxes.

References

[1] J. Lygeros, "On reachability and minimum cost optimal control," *Automatica*, vol. 40, no. 6, pp. 917–927, 2004.

[2] H. Ahn, K. Berntorp, P. Inani, A. J. Ram, and S. Di Cairano, "Reachability-based decision-making for autonomous driving: Theory and experiments," *IEEE Trans. Control Syst. Technol.*, vol. 29, no. 5, pp. 1907–1921, Sep. 2021.

[3] I. Hwang, D. M. Stipanović, and C. J. Tomlin, "Polytopic approximations of reachable sets applied to linear dynamic games and a class of nonlinear systems," in *Advances in Control, Communication Networks, and Transportation Systems*. Boston, MA, USA: Birkhäuser, 2005, pp. 3–19.

[4] P. Varaiya, "Reach set computation using optimal control," in *Verification of Digital and Hybrid Systems*. Berlin, Germany: Springer, 2000, pp. 323–331.

[5] J. A. D. Sandretto and J. Wan, "Reachability analysis of nonlinear odes using polytopic based validated Runge–Kutta," in *Proc. Int. Conf. Reachability Problems*. Cham, Switzerland: Springer, 2018, pp. 1–14.

[6] F. D. Torrisi and A. Bemporad, "HYSDEL—A tool for generating computational hybrid models for analysis and synthesis problems," *IEEE Trans. Control Syst. Technol.*, vol. 12, no. 2, pp. 235–249, Mar. 2004.

[7] M. Althoff, D. Grebenyuk, and N. Kochdumper, "Implementation of Taylor models in CORA 2018," in *Proc. 5th Int. Workshop Appl. Verification Continuous Hybrid Syst.*, 2018, pp. 145–173.

[8] F. Immler et al., "ARCH-COMP18 category report: Continuous and hybrid systems with nonlinear dynamics," in *Proc. 5th Int. Workshop Appl. Verification Continuous Hybrid Syst.*, 2018, pp. 53–70.

[9] D. Romeres, M. Zorzi, R. Camoriano, S. Traversaro, and A. Chiuso, "Derivative-free online learning of inverse dynamics models," *IEEE Trans. Control Syst. Technol.*, vol. 28, no. 3, pp. 816–830, May 2020.

[10] S. Devasia, "Iterative machine learning for output tracking," *IEEE Trans. Control Syst. Technol.*, vol. 27, no. 2, pp. 516–526, Mar. 2017.

[11] Y. Vaupel, N. C. Hamacher, A. Caspari, A. Mhamdi, I. G. Kevrekidis, and A. Mitsos, "Accelerating nonlinear model predictive control through machine learning," *J. Process Control*, vol. 92, pp. 261–270, Aug. 2020.

[12] J. Cheng, J. H. Park, J. Cao, and W. Qi, "Hidden Markov model-based nonfragile state estimation of switched neural network with probabilistic quantized outputs," *IEEE Trans. Cybern.*, vol. 50, no. 5, pp. 1900–1909, May 2020.

[13] O. Thapliyal and I. Hwang, "Learning based cyberattack design and defense for supervisory control systems," in *Proc. Eur. Control Conf. (ECC)*, Jun. 2021, pp. 144–149.

[14] S. Bansal, A. K. Akametalu, F. J. Jiang, F. Laine, and C. J. Tomlin, "Learning quadrotor dynamics using neural network for flight control," in *Proc. IEEE 55th Conf. Decis. Control (CDC)*, Dec. 2016, pp. 4653–4660.

[15] A. Chiuso and G. Pillonetto, "System identification: A machine learning perspective," *Annu. Rev. Control, Robot., Auto. Syst.*, vol. 2, no. 1, pp. 281–304, May 2019.

[16] G. Pillonetto, F. Dinuzzo, T. Chen, G. De Nicolao, and L. Ljung, "Kernel methods in system identification, machine learning and function estimation: A survey," *Automatica*, vol. 50, no. 3, pp. 657–682, Mar. 2014.

[17] A. Garg, K. Tai, and B. Panda, "System identification: Survey on modeling methods and models," in *Artificial Intelligence and Evolutionary Computations in Engineering Systems*. Singapore: Springer, 2017, pp. 607–615.

[18] C. Folkestad and J. W. Burdick, "Koopman NMPC: Koopman-based learning and nonlinear model predictive control of control-affine systems," 2021, *arXiv:2105.08036*.

[19] Y. Han, W. Hao, and U. Vaidya, "Deep learning of Koopman representation for control," in *Proc. 59th IEEE Conf. Decis. Control (CDC)*, Dec. 2020, pp. 1890–1895.

[20] S. Dutta, S. Jha, S. Sankaranarayanan, and A. Tiwari, "Output range analysis for deep feedforward neural networks," in *Proc. NASA Formal Methods Symp.* Cham, Switzerland: Springer, 2018, pp. 121–138.

[21] W. Xiang and T. T. Johnson, "Reachability analysis and safety verification for neural network control systems," 2018, *arXiv:1805.09944*.

[22] C. Huang, J. Fan, W. Li, X. Chen, and Q. Zhu, "ReachNN: Reachability analysis of neural-network controlled systems," *ACM Trans. Embedded Comput. Syst.*, vol. 18, no. 5s, pp. 1–22, Oct. 2019.

[23] W. Xiang, H.-D. Tran, J. A. Rosenfeld, and T. T. Johnson, "Reachable set estimation and safety verification for piecewise linear systems with neural network controllers," in *Proc. Annu. Amer. Control Conf. (ACC)*, Jun. 2018, pp. 1574–1579.

[24] S. Sonoda and N. Murata, "Neural network with unbounded activation functions is universal approximator," *Appl. Comput. Harmon. Anal.*, vol. 43, no. 2, pp. 233–268, Sep. 2017.

[25] J. L. Proctor, S. L. Brunton, and J. N. Kutz, "Dynamic mode decomposition with control," *SIAM J. Appl. Dyn. Syst.*, vol. 15, no. 1, pp. 142–161, 2016.

[26] I. Mezic, "On numerical approximations of the Koopman operator," 2020, *arXiv:2009.05883*.

[27] A. Mauroy, I. Mezić, and Y. Susuki, *The Koopman Operator in Systems and Control: Concepts, Methodologies, and Applications*, vol. 484. Springer, 2020.

[28] M. Ravasi and I. Vasconcelos, "PyLops—A linear-operator Python library for scalable algebra and optimization," *SoftwareX*, vol. 11, Jan. 2020, Art. no. 100361.

[29] C. Sanderson and R. Curtin, "Armadillo: A template-based C++ library for linear algebra," *J. Open Source Softw.*, vol. 1, p 26, 2016.

[30] M. Bergamasco and M. Lovera, "Identification of linear models for the dynamics of a hovering quadrotor," *IEEE Trans. Control Syst. Technol.*, vol. 22, no. 5, pp. 1696–1707, Sep. 2014.

[31] P. Wang, Z. Man, Z. Cao, J. Zheng, and Y. Zhao, "Dynamics modelling and linear control of quadcopter," in *Proc. Int. Conf. Adv. Mech. Syst. (ICAMechS)*, Nov. 2016, pp. 498–503.

[32] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Multistep neural networks for data-driven discovery of nonlinear dynamical systems," 2018, *arXiv:1801.01236*.

[33] O. Thapliyal and I. Hwang, "Approximate reachability for Koopman systems using mixed monotonicity," *IEEE Access*, vol. 10, pp. 84754–84760, 2022.

[34] M. Abate and S. Coogan, "Computing robustly forward invariant sets for mixed-monotone systems," *IEEE Trans. Autom. Control*, pp. 4553–4559, 2022.