Advantages of Bilinear Koopman Realizations for the Modeling and Control of Systems With Unknown Dynamics

Daniel Bruder, *Member, IEEE*, Xun Fu, and Ram Vasudevan, *Member, IEEE*

***Abstract*—Nonlinear dynamical systems can be made easier to control by lifting them into the space of observable functions, where their evolution is described by the linear Koopman operator. This letter describes how the Koopman operator can be used to generate approximate linear, bilinear, and nonlinear model realizations from data, and argues in favor of bilinear realizations for characterizing systems with unknown dynamics. Necessary and sufficient conditions for a dynamical system to have a valid linear or bilinear realization over a given set of observable functions are presented and used to show that every control-affine system admits an infinite-dimensional bilinear realization, but does not necessarily admit a linear one. Therefore, approximate bilinear realizations constructed from generic sets of basis functions tend to improve as the number of basis functions increases, whereas approximate linear realizations may not. To demonstrate the advantages of bilinear Koopman realizations for control, a linear, bilinear, and nonlinear Koopman model realization of a simulated robot arm is constructed from data. In a trajectory following task, the bilinear realization exceeds the prediction accuracy of the linear realization and the computational efficiency of the nonlinear realization when incorporated into a model predictive control framework.**

***Index Terms*—Model learning for control.**

I. Introduction

LINEAR systems theory provides a wealth of analysis and control design techniques for linear dynamical systems, but no such general framework exists for all nonlinear dynamical systems. For this reason, it is common to approximate a nonlinear system with one or more linear systems to make it compatible with well-established and computationally efficient linear control methods.

Linearization of nonlinear systems can be achieved in several ways. Local linearization techniques, such as Jacobian linearization, describe the local behavior near equilibrium points using the first order Taylor series of the dynamics [1]. Such linearizations preserve the stability properties of equilibrium points, but only predict system behavior well in their vicinity.

Global linearization techniques, on the other hand, aim to convert a nonlinear dynamical system into an equivalent linear system. One such technique, based on Koopman operator theory, achieves this by lifting the system into a higher-dimensional space of scalar-valued functions called *observables*. In this space, the evolution of observables along trajectories of the nonlinear system are described by the (linear) Koopman operator [2]. The Koopman operator captures the behavior of the system everywhere, not just near equilibrium points, providing a global linear representation of the system in terms of observables.

Despite their linearity in the space of observables, Koopman representations have limitations that hinder their ability to inform the control of arbitrary nonlinear dynamical systems. One limitation is that linearity with respect to observables does not imply linearity with respect to the control input. Therefore, Koopman models are not necessarily compatible with computationally efficient linear control design techniques such as LQR [3] and linear MPC [4]. A further limitation of Koopman linear embeddings is that they are in general infinite-dimensional. Therefore, finite-dimensional truncations must be used in practice, which merely approximate the behavior of the original nonlinear system rather than capture it perfectly.

Linearity with respect to the control input can be enforced by restricting the Koopman operator to a subspace of observables which only includes linear functions of the control input. This strategy is introduced in [5] to construct linear predictors of nonlinear systems for model predictive control. This and similar approaches have shown capable of achieving better control performance than local linearization techniques on a variety of different robots such as a Sphero SDK [6], a swimming fish robot [7], a Rethink Sawyer robot arm [8], and several soft robots [9], [10].

Such applications make use of finite-dimensional matrix approximations of the Koopman operator that are identified from data using a linear system identification technique known as Extended Dynamic Mode Decomposition (EDMD) [11], [12]. These approximations rely upon the implicit assumption that the identified Koopman matrix converges to the true Koopman operator as its dimension goes to infinity. In other words, they assume that sufficient accuracy can always be achieved by a Koopman model with linear control inputs if its dimension is large enough. However, as shown in [13], a valid Koopman representation is not guaranteed to exist when the approximation is restricted to a subspace that excludes nonlinear functions of the input. This holds true even if the subspace is infinite-dimensional.

Manuscript received October 15, 2020; accepted February 21, 2021. Date of publication March 23, 2021; date of current version April 9, 2021. This letter was recommended for publication by Associate Editor G. Palli and Editor C. Gosselin upon evaluation of the reviewers' comments. This work was supported by the National Science Foundation Career Award #1751093 and the Office of Naval Research under Award Number N00014-18-1-2575. *(Corresponding author: Daniel Bruder.)*

Daniel Bruder is with the School of Engineering and Applied Sciences, Harvard University, Cambridge, MA 02138 USA (e-mail: dbruder@seas.harvard.edu).

Xun Fu and Ram Vasudevan are with the Mechanical Engineering Department, University of Michigan, Ann Arbor, MI 48109 USA (e-mail: xunfu@umich.edu; ramv@umich.edu).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2021.3068117, provided by the authors.

Digital Object Identifier 10.1109/LRA.2021.3068117

There is thus a trade-off when looking to utilize Koopman representations for control of a system with unknown dynamics. Koopman representations with linear control inputs are advantageous because they are compatible with computationally efficient linear control techniques. However, a valid Koopman representation of this form may not exist, in which case finite-dimensional approximations of this type are unlikely to improve with dimension. On the other hand, every system admits a valid Koopman representation with nonlinear control inputs. However, such representations have limited usefulness for control applications because they are not compatible with efficient control techniques.

Koopman representations with bilinear control input terms strike a compromise between these two extremes. Such models preserve some of the computational benefits of linear Koopman models, while being more likely to exist for arbitrary dynamical systems. This makes them desirable for control applications.

Previous work has investigated Koopman-based bilinearization and control methods for nonlinear systems. In [14], the Koopman Canonical Transform (KCT) is used to describe how to construct bilinear realizations over the set of Koopman eigenfunctions. It also introduces a set of sufficient conditions for a nonlinear system to be bilinearizable. In [15], it is shown that the control-affine property of dynamical systems is preserved for the continuous Koopman operator, which allows bilinear surrogate models to be constructed by interpolating between different operators. These bilinear models are incorporated into a model predictive control framework and applied to the control of several nonlinear systems.

In line with previous work, this letter explores the benefits of using the Koopman operator to build bilinear models of systems with unknown dynamics. While approximate Koopman model realizations with linear, bilinear, or nonlinear control inputs can be directly constructed from data, we argue that bilinear Koopman realizations identified in this manner are likely to yield better predictions than linear ones, and be more computationally efficient than nonlinear ones.

The contributions of this letter are twofold. First, we offer a theoretical justification for the proposed advantages of bilinear Koopman model realizations. We specify necessary and sufficient conditions for a dynamical system to have a valid linear or bilinear realization over a given set of observables, and use these conditions to show that every control-affine system admits a bilinear realization, but does not necessarily admit a linear one. Second, we provide instructions on how to construct approximate linear, bilinear, and nonlinear Koopman model realizations from data. Using this approach, we construct several Koopman model realizations of a simulated robot arm, and show that a bilinear realization simultaneously exceeds the prediction accuracy of a linear realization and the computational efficiency of a nonlinear realization when incorporated into a model predictive control framework.

II. Theory

This section describes the Koopman operator and how it relates to model realizations. A definition for linear and bilinear realizations is presented, as well as necessary and sufficient conditions for a realization to be of one of these types. Unpacking these conditions reveals that every control-affine system has a (possibly infinite-dimensional) bilinear realization, but not necessarily a linear one.

A. Koopman Operator Preliminaries

Consider the (nonlinear) dynamical system governed by the following differential equation

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{F}(\boldsymbol{x}(t), \boldsymbol{u}(t)) \tag{1}$$

where $\boldsymbol{x}(t) = [x_1(t), \ldots, x_n(t)]^\top \in X \subset \mathbb{R}^n$ is the state and $\boldsymbol{u}(t) = [u_1(t), \ldots, u_m(t)]^\top \in U \subset \mathbb{R}^m$ is the input of the system at time $t \in [0, +\infty)$, $\boldsymbol{F}$ is a continuously differentiable function, and X, U are compact subsets. Denote by $\phi_\tau : X \times U \to X \times U$ the *flow map*, where $\phi_\tau(\boldsymbol{x}(0), \boldsymbol{u}(0)) = (\boldsymbol{x}(\tau), \boldsymbol{u}(0))$ and $\boldsymbol{x}(\tau)$ is the solution to (1) at time τ when beginning with the initial condition $\boldsymbol{x}(0)$ at time 0 and a constant input $\boldsymbol{u}(0)$ applied for all time between 0 and τ.

Let $\mathcal{F}$ be the infinite-dimensional function space composed of all square-integrable real-valued functions with compact domain $X \times U \subset \mathbb{R}^{n \times m}$. Elements of $\mathcal{F}$ are called *observables*. In $\mathcal{F}$, the flow of the system is characterized by the set of Koopman operators $\mathcal{K}_\tau : \mathcal{F} \to \mathcal{F}$, for each $\tau \geq 0$, which describe the evolution of every observable $f \in \mathcal{F}$ along the trajectories of the system according to the following definition:

$$\mathcal{K}_\tau f = f \circ \phi_\tau, \tag{2}$$

where $\circ$ indicates function composition such that

$$(\mathcal{K}_\tau f)(\boldsymbol{x}(t), \tilde{\boldsymbol{u}}) = f(\boldsymbol{x}(t + \tau), \tilde{\boldsymbol{u}}). \tag{3}$$

for a constant input $\tilde{\boldsymbol{u}}$ over the time interval $[t, t + \tau]$.

The set of Koopman operators is generated by an infinitesimal generator, denoted $\mathcal{K}$, according to the following relation [16, Chapter 11],

$$\mathcal{K}_\tau = e^{\tau \mathcal{K}} \tag{4}$$

The infinitesimal generator describes the time derivative of observables along trajectories of the system

$$\mathcal{K} f = \frac{d}{dt} f \tag{5}$$

such that

$$(\mathcal{K} f)(\boldsymbol{x}(t), \boldsymbol{u}(t)) = \frac{\partial f}{\partial \boldsymbol{x}} \boldsymbol{F}(\boldsymbol{x}(t), \boldsymbol{u}(t)) \tag{6}$$

For a given time-step, $\mathcal{K}_\tau$ is sometimes referred to as the *discrete time* Koopman operator and $\mathcal{K}$ is referred to as the *continuous time* Koopman operator [17].

B. Model Realizations

A model realization of (1) is a dynamical system that generates the same state response $\boldsymbol{x}$ as (1) under any input signal $\boldsymbol{u}$. The Koopman operator can be used to generate model realizations. This is done by choosing a countable set of observables $\{f_i \in \mathcal{F}\}_{i=1}^N$ (where $N \in \mathbb{N} \cup \infty$) from which the original state can be recovered through an inverse mapping $C : \mathbb{R}^N \to \mathbb{R}^n$. Taken together, this set of observables constitutes the *lifted state* of the realization. The evolution of each component of the lifted state is described by applying the continuous Koopman operator, and the output equation of the realization consists of the mapping C:

$$\frac{d}{dt} f_i(t) = \mathcal{K} f_i(t) \qquad \text{for } i = 1, \ldots, N \tag{7}$$

$$\boldsymbol{x}(t) = C\left(f_1(t), \ldots, f_N(t)\right) \tag{8}$$

where $f_i(t)$ is shorthand for $f_i(\boldsymbol{x}(t), \boldsymbol{u}(t))$. We say that the realization is defined over the set of observables $\{f_i\}_{i=1}^N$. Note that model realizations are not unique since they depend on the choice of observables.

The Koopman operator is a linear operator, therefore a realization generated using the Koopman operator is linear with respect to the observables over which it is defined. However, because these observables can be nonlinear functions, such realizations are not necessarily linear with respect to the original state and input. Thus, they may not present any advantages in terms of computational efficiency.

Certain types of realizations are particularly amenable to control design since the control input appears either linearly or bilinearly in them. Before formally defining these realizations, we first define the following subsets of the space of all observables $\mathcal{F}$ for notational convenience:

$$\mathcal{X} := \{f \in \mathcal{F} \,|\, f(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{u}}) = \tilde{x}_i \text{ for some } i = 1, \ldots, n\} \tag{9}$$

$$\mathcal{U} := \{f \in \mathcal{F} \,|\, f(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{u}}) = \tilde{u}_i \text{ for some } i = 1, \ldots, m\} \tag{10}$$

$$\mathcal{Z} := \{f \in \mathcal{F} \,|\, f(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{u}}_1) = f(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{u}}_2)\,, \forall \tilde{\boldsymbol{u}}_1, \tilde{\boldsymbol{u}}_2 \in \mathbb{R}^m\} \tag{11}$$

where $\tilde{\boldsymbol{x}} = [\tilde{x}_1, \ldots, \tilde{x}_n] \in X$ and $\tilde{\boldsymbol{u}} = [\tilde{u}_1, \ldots, \tilde{u}_m] \in U$. In other words, $\mathcal{X}$ is the set of functions that project onto components of the state, $\mathcal{U}$ is the set of functions that project onto components of the input, and $\mathcal{Z}$ is the set of all functions that depend on the state only, including constant functions. With this notation in hand, we define linear and bilinear model realizations as follows:

Definition II.1 (Linear and bilinear model realizations): A model realization of $\dot{\boldsymbol{x}}(t) = \boldsymbol{F}(\boldsymbol{x}(t), \boldsymbol{u}(t))$ over the set of observables $\{z_i \in \mathcal{Z}\}_{i=1}^{N}$ is **bilinear** if there exist sets of coefficients $\{a_{ij} \in \mathbb{R}\}_{i=1,j=1}^{N,N}$, $\{b_{ij} \in \mathbb{R}\}_{i=1,j=1}^{N,m}$, $\{h_{ijk} \in \mathbb{R}\}_{i=1,j=1,k=1}^{N,N,m}$, and $\{c_{ij} \in \mathbb{R}\}_{i=1,j=1}^{n,N}$ such that

$$\frac{d}{dt} z_i(t) = \sum_{j=1}^{N} a_{ij} z_j(t) + \sum_{j=1}^{m} b_{ij} u_j(t) + \sum_{j=1}^{m} \sum_{k=1}^{N} h_{ijk} z_k(t) u_j(t) \tag{12}$$

for $i = 1, \ldots, N$ and

$$x_i(t) = \sum_{j=1}^{N} c_{ij} z_j(t) \tag{13}$$

for $i = 1, \ldots, n$ where $z_i(t)$ is shorthand for $z_i(\boldsymbol{x}(t), \boldsymbol{u}(t))$, $\boldsymbol{x}(t) = [x_1(t), \ldots, x_n(t)]^\top$, and $\boldsymbol{u}(t) = [u_1(t), \ldots, u_m(t)]^\top$. If $h_{ijk} = 0$ for all i, j, k, then the realization is said to be **linear**.

The lifted state of these realizations consists entirely of observables that depend on the state only, and the input appears only in linear and bilinear terms.

As stated previously, realizations are not unique. They depend on the choice of observables over which they are defined. For a particular choice of observables to yield a bilinear or linear realization they must satisfy certain conditions. These conditions are presented in the following theorem.

Theorem II.1 (Necessary and sufficient conditions for linear and bilinear realizations): The realization of the system governed by (1) over a set of observables $\bar{\mathcal{Z}} = \{z_i \in \mathcal{Z}\}_{i=1}^{N}$ is:

1) **Bilinear** if and only if

$$\frac{\partial z_i}{\partial \boldsymbol{x}} \boldsymbol{F} \in \text{span}\left(\bar{\mathcal{Z}} \cup \mathcal{U} \cup \{f \cdot g | f \in \bar{\mathcal{Z}}, g \in \mathcal{U}\}\right) \tag{14}$$

for $i = 1, \ldots, N$ and $\mathcal{X} \subset \text{span}(\bar{\mathcal{Z}})$.

2) **Linear** if an only if

$$\frac{\partial z_i}{\partial \boldsymbol{x}} \boldsymbol{F} \in \text{span}\left(\bar{\mathcal{Z}} \cup \mathcal{U}\right) \tag{15}$$

for $i = 1, \ldots, N$ and $\mathcal{X} \subset \text{span}(\bar{\mathcal{Z}})$.

Note that linear systems are by definition also bilinear.

Proof: The definition for both realizations specifies that the state can be recovered as a linear combination of the observables, i.e.

$$x_i = \sum_{j=1}^{N} c_{ij} z_j, \qquad \text{for } i = 1, \ldots, n \tag{16}$$

$$\Leftrightarrow x_i \in \text{span}\left(\bar{\mathcal{Z}}\right), \qquad \text{for } i = 1, \ldots, n \tag{17}$$

$$\Leftrightarrow \mathcal{X} \subset \text{span}(\bar{\mathcal{Z}}). \tag{18}$$

The conditions specified by (14) and (15) are verified by applying the chain rule to the left hand side of (12):

$$\frac{d}{dt} z_i = \frac{\partial z_i}{\partial \boldsymbol{x}} \frac{d\boldsymbol{x}}{dt} = \frac{\partial z_i}{\partial \boldsymbol{x}} \boldsymbol{F}, \qquad \text{for } i = 1, \ldots, N \tag{19}$$

Plugging (19) into the definition of a bilinear realization from Def. II.1 then yields,

$$\frac{\partial z_i}{\partial \boldsymbol{x}} \boldsymbol{F} = \sum_{j=1}^{N} a_{ij} z_j + \sum_{j=1}^{m} b_{ij} u_j + \sum_{j=1}^{m} \sum_{k=1}^{N} h_{ijk} z_k u_j \tag{20}$$

$$\Leftrightarrow \frac{\partial z_i}{\partial \boldsymbol{x}} \boldsymbol{F} \in \text{span}\left(\bar{\mathcal{Z}} \cup \mathcal{U} \cup \{f \cdot g | f \in \bar{\mathcal{Z}}, g \in \mathcal{U}\}\right) \tag{21}$$

Similarly, plugging (19) into the definition of a linear realization from Def. II.1 yields,

$$\frac{\partial z_i}{\partial \boldsymbol{x}} \boldsymbol{F} = \sum_{j=1}^{N} a_{ij} z_j + \sum_{j=1}^{m} b_{ij} u_j \tag{22}$$

$$\Leftrightarrow \frac{\partial z_i}{\partial \boldsymbol{x}} \boldsymbol{F} \in \text{span}(\bar{\mathcal{Z}} \cup \mathcal{U}) \tag{23}$$

■

Since there are computational benefits to performing control synthesis with linear and bilinear realizations, we would prefer to choose sets of observables that admit a realization of one of these types. However, unless the dynamics of the system are known, it is not possible to verify whether a particular set of observables satisfies the conditions outlined in Theorem II.1. Nevertheless, as we show below, one can prove that for control-affine systems, a realization over $\mathcal{Z}$ is bilinear.

Corollary II.1: If the system governed by (1) is control-affine and a set of observables $\bar{\mathcal{Z}} = \{z_i \in \mathcal{Z}\}_{i=1}^{\infty}$ is a basis of $\mathcal{Z}$, then the realization of the system defined over $\bar{\mathcal{Z}}$ is bilinear. ■

Proof: Omitted for brevity. See [18] for complete proof.

A consequence of the preceding corollary is that every control-affine system has a valid bilinear realization over an infinite set of basis functions. Thus, generic sets of basis functions (i.e. polynomial, Fourier, or radial) can be used as the chosen set of observables in $\bar{\mathcal{Z}}$ since linear combinations of them can represent arbitrary functions in $\mathcal{Z}$. It should be noted that the theorem offers no such guarantee that a system has a valid linear realization, no matter the choice of basis functions. Therefore, realizations defined over a finite set of basis functions of the space of state-dependent observables will converge to a valid bilinear realization as the number of basis functions goes to infinity, but will not necessarily converge to a valid linear realization.

III. Techniques for Data-Driven Realization

The previous section describes how a model realization can be constructed over a set of observables using the Koopman

operator. This section describes how to identify matrix approximations of the Koopman operator from data such that they can be used to produce model realizations that are linear, bilinear, or nonlinear.

A. Approximation of the Koopman Operator From Data

The restriction of the Koopman operator to a finite-dimensional subspace can be represented as a matrix. Using the Extended Dynamic Mode Decomposition (EDMD) algorithm [11], [12], we identify a finite-dimensional matrix approximation of the Koopman operator via linear regression applied to observed data.

We first specify a finite-dimensional subspace as the span of a chosen set of M linearly independent observables $\{\psi_i : X \times U \to \mathbb{R}\}_{i=1}^{M}$. We then define a *lifting function* $\boldsymbol{\psi} : X \times U \to \mathbb{R}^M$ which evaluates each of the observables and stacks them into a vector:

$$\psi(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{u}}) := \begin{bmatrix} \psi_1(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{u}}) & \cdots & \psi_M(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{u}}) \end{bmatrix}^\top, \tag{24}$$

where $\tilde{\boldsymbol{x}} \in X$ and $\tilde{\boldsymbol{u}} \in U$.

To approximate the discrete time Koopman operator from a set of experimental data, we take K discrete measurements in the form of so-called "snapshots" $\{\boldsymbol{p}^{(k)}, \boldsymbol{q}^{(k)}, \boldsymbol{u}^{(k)}\}_{k=1}^{K}$ where

$$\boldsymbol{p}^{(k)} := \boldsymbol{x}(t^{(k)}) \tag{25}$$

$$\boldsymbol{q}^{(k)} := \boldsymbol{x}(t^{(k)} + T_s), \tag{26}$$

$t^{(k)}$ denotes the time corresponding to the k^{th} measurement, $\boldsymbol{u}^{(k)}$ is a constant input applied between $\boldsymbol{p}^{(k)}$ and $\boldsymbol{q}^{(k)}$, and T_s is the sampling period, which is assumed to be identical for all snapshots. Note that consecutive snapshots do not have to be generated by consecutive measurements.

By definition, the action of the discrete time Koopman operator $\mathcal{K}_{T_s}$ advances the value of observables one time-step. Therefore, the best matrix approximation of it in the least-squares sense, denoted $\bar{\mathcal{K}}_{T_s}$, is the minimizer to

$$\min_{\bar{\mathcal{K}}} \sum_{k=1}^{K} \left\| \bar{\mathcal{K}}^\top \boldsymbol{\psi}(\boldsymbol{p}^{(k)}, \boldsymbol{u}^{(k)}) - \boldsymbol{\psi}(\boldsymbol{q}^{(k)}, \boldsymbol{u}^{(k)}) \right\|_2^2. \tag{27}$$

Note that the control input $\boldsymbol{u}^{(k)}$ describes the constant input value held *between* $t^{(k)}$ and $t^{(k)} + T_s$, not the instantaneous input applied at either time instance.

With an approximation of the discrete time Koopman operator $\bar{\mathcal{K}}_{T_s}$ in hand, we can solve for the corresponding continuous time Koopman operator $\bar{\mathcal{K}}$ by inverting (4):

$$\bar{\mathcal{K}} = \frac{1}{T_s} \log \mathcal{K}_{T_s} \tag{28}$$

where $\log$ denotes the principal matrix logarithm [16, Chapter 11].

B. Linear Model Realization

The Koopman matrix $\bar{\mathcal{K}}$ on the subspace spanned by a set of observables $\{\psi_i : X \times U \to \mathbb{R}\}_{i=1}^{N+m}$ can be constructed such that it is decomposable into a linear system representation. One way to achieve this is to define the first N basis functions as functions of the state only, and the last m basis functions as projections onto each component of the input, i.e.

$$\psi_i := \begin{cases} z_i & \in \mathcal{Z}, \quad \text{for } i = 1, \ldots, N \\ \pi_{u_{i-N}} & \in \mathcal{U}, \quad \text{for } i = N+1, \ldots, N+m \end{cases} \tag{29}$$

where π_{u_i} denotes the projection onto the i^{th} component of the input. This choice ensures that the input only appears in the last m components of the lifted vector $\boldsymbol{\psi}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{u}})$.

The Koopman matrix can be identified from data according the steps laid out in Section III-A. Then, by construction, the coefficients for a linear realization of the form specified in (12) are embedded within the first N rows of the transpose of the Koopman matrix in the following manner,

$$\bar{\mathcal{K}}^\top = \begin{bmatrix} A_{\{N \times N\}} & B_{\{N \times m\}} \\ \vdots & \vdots \end{bmatrix} \tag{30}$$

where

$$A = \begin{bmatrix} a_{11} & \cdots & a_{1N} \\ \vdots & \ddots & \vdots \\ a_{N1} & \cdots & a_{NN} \end{bmatrix}, \quad B = \begin{bmatrix} b_{11} & \cdots & b_{1m} \\ \vdots & \ddots & \vdots \\ b_{N1} & \cdots & b_{Nm} \end{bmatrix} \tag{31}$$

and the subscripts in curly brackets denote the dimensions of each matrix.

For convenience, we can define the first n basis functions as projections onto each component of the state, i.e.

$$\psi_i := \pi_{x_i} \in \mathcal{X} \subset \mathcal{Z} \quad \text{for } i = 1, \ldots, n \tag{32}$$

Then, the coefficients of the output equations of (13) can be defined as simply

$$c_{ij} = \begin{cases} 1, & \text{for } i = j \\ 0, & \text{for } i \neq j \end{cases} \tag{33}$$

C. Bilinear Model Realization

With a suitable choice of observables $\{\psi_i\}_{i=1}^{N(m+1)+m}$, the Koopman matrix $\bar{\mathcal{K}}$ can be constructed such that it is decomposable into a bilinear system representation. The first N observables are defined as functions of the state only, the next Nm observables are defined as the product of the first N basis functions and each component of the input, and the last m basis functions are defined as projections onto each component of the input, i.e.

$$\psi_i := \begin{cases} z_i \in \mathcal{Z}, & \text{for } i = 1, \ldots, N \\ z_{i-N} \cdot \pi_{u_1}, & \text{for } i = N+1, \ldots, 2N \\ \quad \vdots & \quad \vdots \\ z_{i-Nm} \cdot \pi_{u_m}, & \text{for } i = Nm+1, \ldots, N(m+1) \\ \pi_{u_{i-N(m+1)}} \in \mathcal{U}, & \text{for } i = N(m+1)+1, \ldots, \\ & \qquad N(m+1)+m \end{cases} \tag{34}$$

The Koopman matrix can be identified from data according the steps laid out in Section III-A. Then, by construction, the coefficients for a bilinear realization of the form specified in (12) are embedded within the first N rows of the transpose of the Koopmn matrix in the following manner,

$$\bar{\mathcal{K}}^\top = \begin{bmatrix} A & H_1 & \cdots & H_m & B \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix} \tag{35}$$

where A and B are defined as in (31) and

$$H_i = \begin{bmatrix} h_{i11} & \cdots & h_{i1N} \\ \vdots & \ddots & \vdots \\ h_{iN1} & \cdots & h_{iNN} \end{bmatrix}. \tag{36}$$

Just as with the linear model realization, we can define the first n basis functions as in (32), and define the coefficients of the output equation by (33)

D. Nonlinear Model Realization

If the Koopman matrix $\bar{\mathcal{K}}$ is identified on a subspace spanned by a set observables $\{\psi_i : X \times U \to \mathbb{R}\}_{i=1}^{M}$ other than the type specified in Sections III-B and III-C, then the realization it produces will be nonlinear, i.e. it may contain nonlinear functions of the input. Such a representation takes the form of a linear combination of the basis functions for each component of the state:

$$x_i = \sum_{j=1}^{M} c_{ij}\psi_j, \qquad \text{for } i = 1, \ldots, n \tag{37}$$

Assuming once again that the first n observables are defined as projections onto each component of the state as in (32), by construction, the coefficients are embedded within the first n rows of the transpose of the Koopman matrix, i.e.

$$\bar{\mathcal{K}}^{\top} = \begin{bmatrix} c_{11} & \cdots & c_{1M} \\ \vdots & \ddots & \vdots \\ c_{n1} & \cdots & c_{nM} \\ \vdots & \vdots & \vdots \end{bmatrix} \tag{38}$$

E. Model Predictive Control

Model predictive control algorithms optimally choose a sequence of control inputs given a desired output trajectory and system model [4]. This system model can be either linear or nonlinear, but linear models have computational advantages over nonlinear ones. Namely, the MPC optimization problem for linear models is convex, while for nonlinear models it is not.

Since the linear MPC optimization problem is convex, it has a unique globally optimal solution that can be efficiently computed without initialization even for high-dimensional models [19]–[21]. This is not the case for the nonlinear MPC problem which has nonlinear constraints that render it non-convex [22]. As a result, algorithms to solve such problems typically require initialization, can struggle to find globally optimal solutions [23], and take longer to solve per iteration. Though techniques have been proposed to improve the speed and accuracy of nonlinear MPC algorithms [24]–[26] they are still often unable to achieve the computational efficiency required for real-time control.

In this letter we implement three model predictive controller algorithms which we refer to by the following abbreviations:

- K-MPC: Koopman-based linear MPC
- K-BMPC: Koopman-based bilinear MPC
- K-NMPC: Koopman-based nonlinear MPC

The K-MPC and K-NMPC controllers are standard model predictive controllers that use linear and nonlinear Koopman model realizations to generate predictions, respectively. Similarly, the K-BMPC controller uses a bilinear Koopman model realization to generate predictions. However, bilinear constraints would render the problem non-convex, so we instead rely on a linear approximation constructed by fixing the value of the lifted state in the bilinear terms over the prediction horizon, N_h. The dynamics constraints in the K-BMPC optimization problem then become:

$$z_i[t+1] = \sum_{j=1}^{N} a_{ij} z_j[t] + \sum_{j=1}^{m} b_{ij} u_j[t] + \sum_{j=1}^{m}\sum_{k=1}^{N} h_{ijk} z_k[0] u_j[t] \tag{39}$$

$$= \sum_{j=1}^{N} a_{ij} z_j[t] + \sum_{j=1}^{m} \left(b_{ij} + \sum_{k=1}^{N} h_{ijk} z_k[0] \right) u_j[t] \tag{40}$$

for $i = 1, \ldots, N$ and $t = 1, \ldots, N_h$, where $[\cdot]$ denotes a discrete time index. The resulting linear dynamics approximate the behavior of the bilinear realization in a neighborhood of the initial lifted state $\{z_i[0]\}_{i=1}^{N}$. This introduces prediction error, but turns it into a convex quadratic program which can be solved quickly enough to be updated at every time-step.

Beyond the dynamics constraints, all three MPC controllers can accommodate additional linear state and input constraints. Constraints on nonlinear functions of the unlifted state $\boldsymbol{x}$ can be included as linear constraints as long as such functions are included in the set of observables over which a realization is defined. In this way, nonlinear constraints on $\boldsymbol{x}$ can be imposed while still preserving the convexity of the K-MPC and B-MPC optimization problems.

The main difference between the K-BMPC controller and the other MPC algorithms is that it does not generate optimal solutions with respect to the model upon which it is based. Instead, the solutions it generates are optimal with respect to a linear approximation of the actual bilinear MPC problem. It is important to note, however, that all finite-dimensional Koopman realizations constructed from data are mere approximations of the true dynamics of a system. Hence, even the optimal solutions to the linear/nonlinear MPC problems are not necessarily optimal with respect to the true dynamics of the system.

IV. Experiments and Results

To evaluate the relative performance of linear, bilinear, and nonlinear Koopman realizations for modeling systems with unknown dynamics, we applied the methods from Section III to a collection of randomly generated dynamical systems as well as a simulated planar arm system. The code used to generate these results can be found in a publicly accessible repository.[1]

A. Description of Systems

20 systems with one-dimensional input $u(t) \in \mathbb{R}$ and state $x(t) \in \mathbb{R}$ were generated with dynamics of the following form,

$$\dot{x}(t) = e^{-x(t)^4} \left(\sum_{i=1}^{3} \lambda_i x(t)^{\mu_i} u(t)^{\sigma_i} + \lambda_4 u(t) \right) - \tan^{-1}(x(t)) \tag{41}$$

where the coefficients $\lambda_i \sim \text{unif}(-1, 1)$ and the exponents $\mu_i, \sigma_i \sim \text{unif}\{0, 1, 2\}$ for each system were selected from the continuous and discrete uniform distributions, respectively. Note that these systems are not strictly control-affine, since higher order input terms are permitted. The exponential and inverse tangent terms ensure the state of each system remains finite under arbitrary inputs while still allowing for large variability in behavior. The data used to identify Koopman realizations

[1] [Online]. Available: https://github.com/roahmlab/koopman-realizations.git

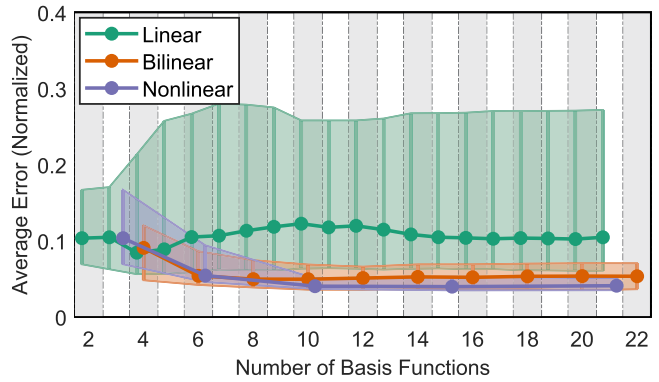

Fig. 1. The model prediction error versus the number of basis functions for several linear, bilinear, and nonlinear realizations of 20 randomly generated systems. Dots indicate the median prediction error and the lower and upper bounds of the shaded regions signify the lower and upper quartiles, respectively.

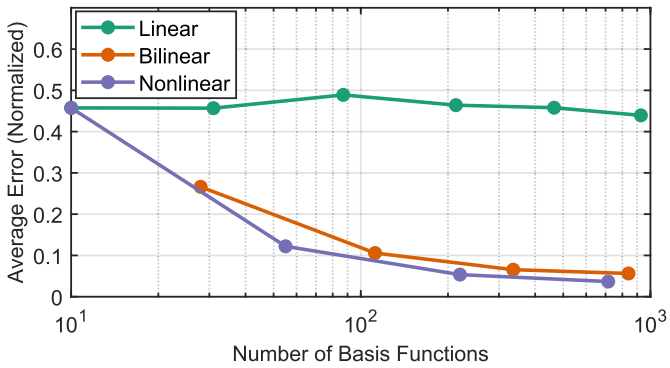

Fig. 2. The model prediction error for several linear, bilinear, and nonlinear Koopman model realizations of the simulated arm system. As the number of basis functions increases, the error of the linear model changes little while the error of the bilinear and nonlinear models decrease monotonically.

of each of the systems was a set of 10 000 snapshots with a time-step of $T_s = 0.01$s collected over a range of randomized initial conditions and inputs.

In addition to these randomly generated systems, a simulated planar arm system was also considered (Fig. 3). The arm has 3-links each of mass 100 g and length 0.33 m and 3 joints each with a stiffness of 1×10^{-5} N/rad and a viscous damping coefficient of 1 Ns/rad. The input into the system is a set of $m = 3$ applied joint torques and the output is the location of the end of each link expressed as a $n = 6$ dimensional vector of Cartesian coordinates,

$$\boldsymbol{u}(t) = [\tau_1(t),\, \tau_2(t),\, \tau_3(t)]^\top \tag{42}$$

$$\boldsymbol{x}(t) = [\alpha_1(t),\, \beta_1(t),\, \alpha_2(t),\, \beta_2(t),\, \alpha_3(t),\, \beta_3(t)]^\top \tag{43}$$

The data used to identify Koopman realizations of the arm system was a set of 12 000 snapshots with a time-step of $T_s = 0.05$s collected over a range of randomized initial conditions and inputs.

B. Model Prediction Comparison

The predictive accuracy of various models identified using the Koopman approach depends on both the model type (e.g. linear, bilinear, or nonlinear) as well as the number of basis functions. We identified several models of each type on subspaces spanned by monomial basis functions.

TABLE I
NUMBER OF MONOMIAL BASIS FUNCTIONS FOR MODELS OF ARM

ρ	# of basis functions, i.e. M		
	Linear	Bilinear	Nonlinear
1	10	28	10
2	31	112	55
3	87	336	220
4	213	840	715
5	465	-	-
6	927	-	-

The sets of basis functions used for identifying the linear models consisted of all monomials of the components of the state up to a specific degree denoted ρ, plus the projections onto each component of the input, i.e.

$$\{\psi_i(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{u}})\}_{i=1}^{M} = \{\tilde{x}_1^{\rho_1} \cdots \tilde{x}_n^{\rho_n} | \rho_1 + \cdots + \rho_n \leq \rho\} \cup \{\tilde{u}_i | i \in \{1, \ldots, m\}\} \tag{44}$$

where $M = N + m$ and $N = (n+\rho)!/(n!\rho!)$. The sets of basis functions used for identifying bilinear models consisted of the same monomials as well as the product of each monomial with each component of the input, i.e.

$$\{\psi_i(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{u}})\}_{i=1}^{M} = \{(\tilde{x}_1^{\rho_1} \cdots \tilde{x}_n^{\rho_n})\hat{u} | \rho_1 + \cdots + \rho_n \leq \rho, \hat{u} \in \{1, \tilde{u}_1, \ldots, \tilde{u}_m\}\} \tag{45}$$

where $M = (N+1)(m+!)$ and $N = (n+\rho)!/(n!\rho!)$. The sets of basis functions used for identifying nonlinear models consisted of monomials up to degree ρ of both the input and the state, i.e.

$$\{\psi_i(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{u}})\}_{i=1}^{M} = \left\{(\tilde{x}_1^{\rho_1} \cdots \tilde{x}_n^{\rho_n})(\tilde{u}_1^{\rho_{n+1}} \cdots \tilde{u}_m^{\rho_{n+m}}) | \rho_1 + \cdots + \rho_{n+m} \leq \rho\right\} \tag{46}$$

where $M = (n+m+\rho)!/((n+m)!\rho!)$.

For each of the 20 randomly generated systems, 20 linear models were identified for values of $\rho = \{1, .., 20\}$, 10 bilinear models were identified for values of $\rho = \{1, .., 10\}$, and 5 nonlinear models were identified for values of $\rho = \{1, .., 5\}$. The prediction accuracy of each model was evaluated by comparing a model simulation to validation data for a 10 s trial, computing the average error over all time-steps, then normalizing by the average error incurred by the zero response over all time-steps. Fig. 1 displays the median prediction error over all random systems versus the number of basis functions used to identify the Koopman operator matrix, i.e. $\dim(\boldsymbol{\psi}(\tilde{x}, \tilde{u}))$. The median prediction error of the bilinear and nonlinear realizations generally decreases then plateaus as the number of basis functions increases, whereas it varies more erratically for the linear realizations.

For the planar arm system, 6 linear models were identified for values of $\rho = \{1, .., 6\}$, and 4 bilinear and nonlinear models were identified for values of $\rho = \{1, \ldots, 4\}$. Table I indicates the total number of basis functions, used for identifying the Koopman operator matrix for each model. The prediction accuracy of each model was evaluated by comparing model simulations to validation data for a 20 s trial and computing the average error over all time-steps. This error was quantified as the Euclidean distance between the predicted and real outputs in $\mathbb{R}^6$, normalized by the average Euclidean distance error incurred by the zero response over all time-steps.

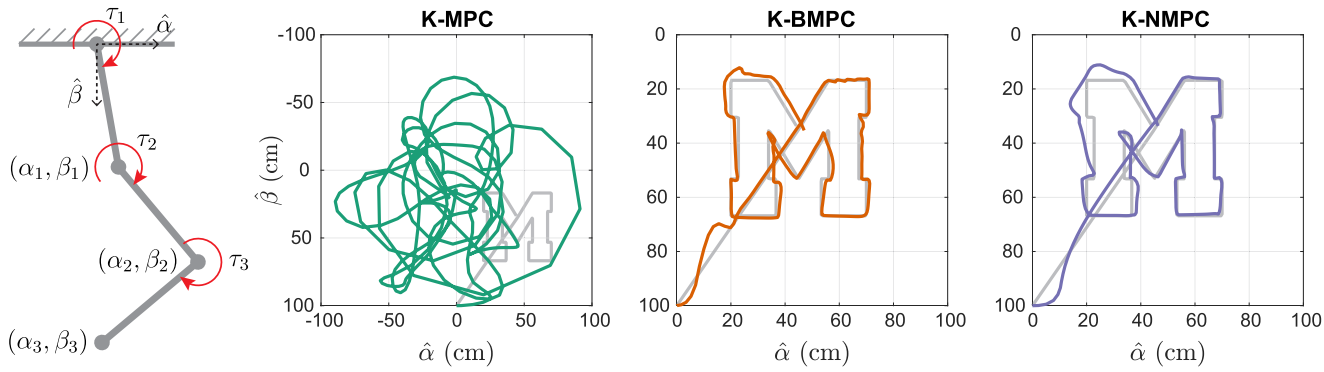

Fig. 3. Three link planar arm system with input defined as joint torques and output defined as the locations of link tips (left). The end effector trajectories generated by the K-MPC (middle-left), K-BMPC (middle-right), and K-NMPC controllers (right) are superimposed over a reference trajectory, shown in grey.

Fig. 2 displays the prediction error versus the dimension of the Koopman operator matrix for each model. The accuracy of the linear model increases very little, even when the dimension of the system is increased by several orders of magnitude. The bilinear and nonlinear models become more accurate as the dimension increases.

C. Control Performance Comparison

To highlight the relative strengths and weaknesses of the Koopmans-based control techniques described in Section IV-B, we applied them to the simulated 3-link planar arm system. A K-MPC controller, K-BMPC controller, and K-NMPC controller was constructed from the linear, bilinear, and nonlinear model realizations identified in Section IV-B for $\rho = 3$, respectively. Each controller was then employed to perform the same trajectory following task. Video of this experiment can be found in a supplementary video file.[2]

Each controller computed solutions over a $N_h = 10$ step horizon and had identical cost functions. The desired task was to move the end effector of the arm along a planar reference trajectory. Therefore, a cost function was chosen that penalizes the distance between the actual end effector coordinates (α_3, β_3) and the desired coordinates $(\alpha_3^{\text{ref}}, \beta_3^{\text{ref}})$ as well as the control effort at each time-step.

The control experiment was conducted in simulation. At each time-step, the current output of the system is measured and used to initialize the MPC optimization problem. Once a solution is computed, the system is simulated forward one time-step ($T_s = 0.05$ seconds) under the optimal input. This procedure is repeated until the end of the reference trajectory is reached.

The reference trajectory traces out the shape of a block letter M over a time period of 15 seconds, starting from the robot's hanging position. Fig. 3 shows the path of the end effector using each of the controllers, and Fig. 4 displays the mean tracking error and mean computation time over all time-steps. The tracking error at each time-step is quantified as the Euclidean distance between the actual and desired end effector locations. The computation time per iteration is the amount of time it takes to solve the MPC optimization problem and does not include the time to simulate the response of the system. All three trials were run on a computer with 64 GB RAM and a 2.4 GHz CPU.

[2][Online]. Available: https://youtu.be/F-vJoBbAdJE

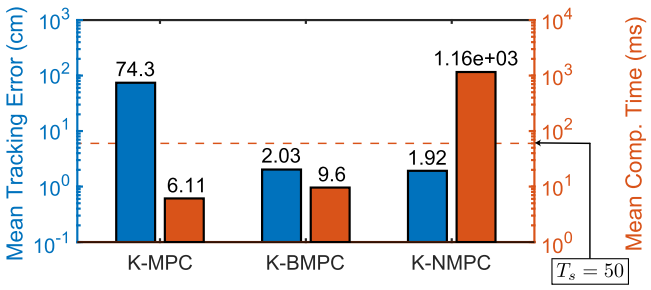

Fig. 4. The mean tracking error (blue) and the mean computation time (orange) for each controller plotted on a logarithmic scale. The K-BMPC controller has a mean tracking error comparable to K-NMPC and a mean computation time comparable to K-MPC, proving it to be both accurate and computationally efficient.

V. Discussion and Conclusion

The results of the model prediction comparisons described in Section IV-B illustrate the advantages of bilinear realizations for systems with unknown dynamics. As seen in in both Fig. 1 and Fig. 2, as the number of basis functions increases, the prediction error of the bilinear model realizations decreases, indicating progress toward a true infinite-dimensional bilinear realization. The linear model realizations, on the other hand, do not consistently improve with the inclusion of more basis functions, indicating that an infinite-dimensional linear realization over monomial basis functions probably doesn't exist.

This phenomenon emerges even for dynamical systems, like those governed by (41), which are known not to be control-affine, as shown in Fig. 1. The bilinear realizations outperform the linear realizations in terms of both accuracy and consistency, as indicated by their lower median error and narrower quartile range. In Section II, we only proved the existence of infinite-dimensional bilinear realizations for control-affine systems, but these results suggest bilinear realizations may exist for a wider class of systems. This should be a topic of further study.

Bilinear Koopman realizations have benefits when it comes to control as well. It is clear by inspection of Fig. 3 that the K-MPC controller performs very poorly. This is confirmed quantitatively in Fig. 4 which shows that its mean tracking error is more than 15$\times$ larger than that of the other controllers. This poor performance can be attributed to the inaccuracy of the linear Koopman model realization upon which it is based, which is documented in Fig. 2. The K-BMPC and K-NMPC controllers

track the desired trajectory with much greater fidelity, reflecting the greater accuracy of the bilinear and nonlinear model realizations upon which they are based.

In Section II-B, we asserted that K-NMPC is much less computationally efficient than the other controllers, and that is confirmed by the results of this experiment. As seen in Fig. 4, the mean computation time for K-NMPC is more than $100\times$ larger than that of the other two controllers. This computation time greatly exceeds the 50 ms duration of a single time-step, making it incompatible with closed-loop operation. Hence, if this robot were a real physical system, the control inputs would have to be computed offline.

Based on the results of this experiment, only K-BMPC would be a viable closed-loop controller for this system. Its mean computation time is much less than a single time-step, and despite the suboptimality of its solutions, its mean tracking error is nearly equivalent to that of K-NMPC. Roughly speaking, K-MPC fast but inaccurate, K-NMPC is accurate but slow, and K-BMPC is both fast and accurate.

This works shows that when the dynamics of a system are completely unknown, a bilinear Koopman realization constructed from data is likely to yield better overall modeling and control results than a linear or nonlinear Koopman realization. This is justified theoretically in Section II and demonstrated practically in Section IV. We proved that control-affine systems have infinite-dimensional bilinear realizations but not necessarily linear ones. Therefore, approximate bilinear realizations constructed from generic sets of basis functions improve as the number of basis functions increases, whereas approximate linear realizations may not.

Bilinear realizations combine the computational efficiency of linear realizations with the prediction accuracy of nonlinear realizations. However, the bilinear model predictive control framework used in this letter could likely be improved. Other techniques have been successfully applied to solve optimal control problems with bilinear constraints in [27], [28] without relying on a linear approximation at each time-step. Further work should compare the performance of these existing bilinear optimal control approaches, investigate theoretical guarantees for bilinear controllers, and explore new approaches to optimal control of bilinear dynamical systems.

REFERENCES

[1] K. J. Astrom and R. M. Murray, *Feedback Systems: An Introduction for Scientists and Engineers*. Princeton, NJ, USA: Princeton Univ. Press, 2010.

[2] M. Budisic, R. Mohr, and I. Mezic, "Applied koopmanism," Chaos: *An Interdiscipl. J. Nonlinear Sci.*, vol. 22, no. 4, 2012, Art. no. 0 47510.

[3] B. D. Anderson and J. B. Moore, *Optimal Control: Linear Quadratic Methods. Courier Corporation*, Mineda, New York, USA: Dover Publications Inc., 2007.

[4] J. B. Rawlings and D. Q. Mayne, *Model Predictive Control: Theory and Design*. Madison, Wisconsin, WI, USA: Nob Hill Pub., 2009.

[5] M. Korda and I. Mezic, "Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control," *Automatica*, vol. 93, pp. 149–160, 2018.

[6] I. Abraham, G. De La Torre, and T. D. Murphey, "Model-based control using koopman operators," 2017, *arXiv:1709.01568*.

[7] G. Mamakoukas, M. Castano, X. Tan, and T. Murphey, "Local koopman operators for data-driven control of robotic systems," in *Robot.: Sci. Syst.*, Jun. 2019, doi: 10.15607/RSS.2019.XV.054.

[8] I. Abraham and T. D. Murphey, "Active learning of dynamics for data-driven control using koopman operators," *IEEE Trans. Robot.*, vol. 35, no. 5, pp. 1071–1083, Oct. 2019.

[9] D. Bruder, B. Gillespie, C. D. Remy, and R. Vasudevan, "Modeling and control of soft robots using the koopman operator and model predictive control," in *Proc. Robot.: Sci. Syst.*, Jun. 2019, doi: 10.15607/RSS.2019.XV.060.

[10] D. Bruder, X. Fu, R. B. Gillespie, C. D. Remy, and R. Vasudevan, "Koopman-based control of a soft continuum manipulator under variable loading conditions," 2020, *arXiv:2002.01407*.

[11] M. O. Williams, I. G. Kevrekidis, and C. W. Rowley, "A data-driven approximation of the koopman operator: Extending dynamic mode decomposition," *J. Nonlinear Sci.*, vol. 25, no. 6, pp. 1307–1346, 2015.

[12] A. Mauroy and J. Goncalves, "Koopman-based lifting techniques for nonlinear systems identification," *IEEE Trans. Autom. Control*, vol. 65, no. 6, pp. 2550-2565, Jun, 2020.

[13] C. Bakker, S. Rosenthal, and K. E. Nowak, "Koopman representations of dynamic systems with control," 2019, *arXiv:1908.02233*.

[14] D. Goswami and D. A. Paley, "Global bilinearization and reachability analysis of control-affine nonlinear systems," in *The Koopman Operator Syst. Control.*, 2020, pp. 81–98.

[15] S. Peitz, S. E. Otto, and C. W. Rowley, "Data-driven model predictive control using interpolated koopman generators," *SIAM J. Appl. Dynamical Syst.*, vol. 19, no. 3, pp. 2162–2193, 2020.

[16] N. J. Higham, *Functions of Matrices: Theory and Computation*. Philadelphia, PA, USA: Siam, 2008.

[17] S. L. Brunton, B. W. Brunton, J. L. Proctor, and J. N. Kutz, "Koopman invariant subspaces and finite linear representations of nonlinear dynamical systems for control," *PLoS one*, vol. 11, no. 2, 2016, Art. no. e0150171.

[18] D. Bruder, X. Fu, and R. Vasudevan, "Advantages of bilinear koopman realizations for the modeling and control of systems with unknown dynamics," 2020, *arXiv:2010.09961*.

[19] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[20] J. A. Paulson, A. Mesbah, S. Streif, R. Findeisen, and R. D. Braatz, "Fast stochastic model predictive control of high-dimensional systems," in *Proc. IEEE 53rd Annu. Conf. Decis. Control.*, 2014, pp. 2802–2809.

[21] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, "Osqp: An operator splitting solver for quadratic programs," in *Proc. UKACC 12th Int. Conf. Control.*, 2018, pp. 339–339.

[22] F. Allgower and A. Zheng, *Nonlinear Model Predictive Control*. Cambridge, MA, USA: Birkhauser, 2012.

[23] E. Polak, *Optimization: Algorithms and Consistent Approximations*. Berlin, Germany: Springer Science & Business Media, 2012.

[24] M. A. Patterson and A. V. Rao, "Gpops-ii: A matlab software for solving multiple-phase optimal control problems using hp-adaptive gaussian quadrature collocation methods and sparse nonlinear programming," *ACM Trans. Math. Softw.*, vol. 41, no. 1, pp. 1–37, 2014.

[25] A. Hereid and A. D. Ames, "Frost: Fast robot optimization and simulation toolkit," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2017, pp. 719–726.

[26] P. Zhao, S. Mohan, and R. Vasudevan, "Control synthesis for nonlinear optimal control via convex relaxations," in *Proc. Amer. Control Conf.*, 2017, pp. 2654–2661.

[27] A. Surana, "Koopman operator based observer synthesis for control-affine nonlinear systems," in *Proc. IEEE 55th Conf. Decis. Control.*, 2016, pp. 6492–6499.

[28] X. Ma, B. Huang, and U. Vaidya, "Optimal quadratic regulation of nonlinear system using koopman operator," in *Proc. Amer. Control Conf.*, 2019, pp. 4911–4916.