Koopman Operator Based Predictive Control With a Data Archive of Observables[1]

Kartik Loya [*] **Jake Buzhardt** [*] **Phanindra Tallapragada** [*]

[*] *Department of Mechanical Engineering, Clemson University,Clemson, SC 29634 USA (e-mail: kloya@g.clemson.edu, jbuzhar@g.clemson.edu, ptallap@clemson.edu).*

Abstract:
The control of complex systems is often challenging due to high dimensional nonlinear models, unmodeled phenomena, and parameter uncertainty. The increasing ubiquity of sensors measuring such systems and increased computational resources has led to an interest in purely data-driven control methods, particularly using the Koopman operator. In this paper, we elucidate the construction of a linear predictor based on a sequence of time realizations of observables drawn from a data archive of different trajectories combined with subspace identification methods for linear systems. This approach is free of any predefined set of basis functions but instead depends on the time realization of these basis functions. The prediction and control are demonstrated with examples. The basis functions can be constructed using time-delayed coordinates of the outputs, enabling the application to purely data-driven systems. The paper thus shows the link between Koopman operator-based control methods and classical subspace identification methods. The approach in this paper can be extended to adaptive online learning and control.

1. INTRODUCTION

Controlling complex autonomous systems, such as swimming robots, off-road vehicles, wind turbines, transportation networks, and power grids poses significant challenges due to high-dimensional nonlinear models, unmodeled phenomena, and parameter uncertainties. The increasing ubiquity of sensors has led to an abundance of data from such systems, which combined with improved computational resources has led to an interest in purely data-driven methods for modeling, analysis, and control of such systems [1–3]. These data-driven approaches have increasingly been based on linear operators associated with nonlinear dynamical systems. The Perron-Frobenius operator propagates the densities in the state space of a dynamical system, and the Koopman operator propagates the observables along the flow in the state space of a dynamical system, see for example [4]. By considering the evolution of functions of the states rather than the states themselves, these operator-based methods transform a nonlinear system into an infinite dimensional linear system, which can be approximated by a finite-dimensional linear system such as in [5,6]. Additionally, many efforts have been made recently to extend these results from autonomous dynamical systems to input-output systems,

where the operator approximation must account for the effect of a control input to predict the output variables. For example, extended dynamic mode decomposition (EDMD) coupled with model predictive control (MPC) such as in [7] or with linear parameter varying (LPV) modes such as in [8] and variations thereof have become popular. More recently, ideas from subspace identification (SSID) for linear systems, such as from [9] and [10] have been used by [11,12] to develop a method based on identifying the significant modes in the evolution of an observable along a single trajectory and then using Gaussian processes as a mapping to this identified lifted state. The current paper builds on these ideas of subspace identification and clarifies and extends them to include observation data from multiple trajectories and further shows that basis functions for the observables can be defined using only data from output variables.

The main contributions of the paper are as follows. We demonstrate the relation between a Koopman operator associated with a nonlinear system and subspace identification methods of linear systems. We identify the relation between the approximate Koopman operator that is calculated using techniques like EDMD versus subspace methods. In the approach using subspace methods, the approximate Koopman operator that is calculated propagates time realizations of observables. Therefore, output data from a single trajectory does not (usually) sample the state space dynamics, as it is difficult to ensure that a nonlinear system is sufficiently excited, an assumption that underlies the subspace identification methods with a

single data record, see for example [13]. We show that using multiple data records improves the accuracy of prediction. Multiple data records have been shown to be useful even in subspace identification for linear systems in [14] since a sufficiently long single data record may not be available in practice and, more importantly, the observables could be projected on any set of basis functions, such as Gaussian processes. Usually, the basis functions are functions of the state, and in purely data-driven systems, such functions cannot be constructed since explicit knowledge of the states is not available nor are different observables realized or sampled. This paper shows that the basis functions can be constructed using time-delayed coordinates of the outputs, enabling the application to purely data-driven systems.

The rest of the paper is organized as follows. In section 2 we describe the approximations of the different 'versions' of the Koopman operator and in particular the approximation that propagates time realizations of the observables. In section 3 we formulate the subspace identification problem for the linearized dynamics using multiple data records; in 4 we review Gaussian processes (GP) as basis functions (of state variable) for observables and in 4.2 we reformulate the GP as basis functions of time delayed observables and demonstrate numerical results in 6.

2. FORMULATION FOR KOOPMAN OPERATOR BASED CONTROL

Consider the autonomous dynamical system defined on a manifold $\mathcal{M}$

$$\dot{x} = f(x) \tag{1}$$

with $f : \mathcal{M} \mapsto T_x \mathcal{M}$ and suppose the bounded function $g : \mathcal{M} \mapsto \mathbb{R}$ is an observable. The flow map of the dynamical system, $F_t : \mathcal{M} \mapsto \mathcal{M}$ propagates the state as $x(t + t_0) = F_t(x(t_0))$. The Koopman operator $\mathcal{K} : \mathcal{L}^\infty \mapsto \mathcal{L}^\infty$ propagates the observable as $\mathcal{K}g(x(t_0)) = g \circ F_t(x(t_0)))$. For the discrete-time dynamical system

$$x_{t+1} = F(x_t) \tag{2}$$

where $x_t \in \mathcal{M}$ and $F : \mathcal{M} \mapsto \mathcal{M}$, the action of the Koopman operator on an observable function $g : \mathcal{M} \mapsto \mathbb{R}$ is given as follows.

$$\mathcal{K}g(x_t) = (g \circ F)(x_t) = g(x_{t+1}). \tag{3}$$

In practice, the action of the infinite-dimensional Koopman operator is approximated by its action on a finite-dimensional space called the lifted space. This has frequently been done through by projection to a finite set of basis functions called the dictionary [5,7] that can either be defined a priori or learned as part of the system identification procedure or through the use of neural networks such as in [15,16]. Here we take a different approach and treat the sequence of measurements $y_t \in \mathbb{R}^p$ as evaluations of an observable function of the state. This observable function is treated as a linear combination of finite basis functions $\Psi(x_t)$, which would be propagated by an approximation of the Koopman operator K_Ψ

$$\mathcal{K}g(x_t) \approx \mathcal{K}[c^\intercal \Psi](x_t) \approx (K_\Psi c)^\intercal \Psi(x_t) \tag{4}$$

as shown in Fig. 1. Here c is a vector of projection coefficients for the projection of g onto the function space spanned by the dictionary functions Ψ. In order to obtain an appropriate set of dictionary functions, Ψ, we first
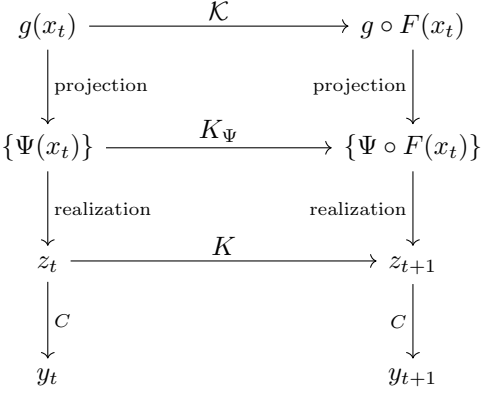

Fig. 1. Commutative diagram showing the propagation of the observable $g(x)$ by the operator $\mathcal{K}$, the propagation of the projected observables $\Psi(x)$ by the projected operator K_Ψ and their time realization z_t by the operator K.

obtain a lifted state $z_t \in \mathbb{R}^r$ and a corresponding operator K, which propagates it forward in time by applying ideas of subspace identification using temporal sequences of data without first approximating the operator K_Ψ or specifying the basis functions Ψ. These lifted states z_t are then regarded as evaluations at a particular x_t of a realization of a random variable in a function space. This notion of a random variable in a function space allows us to apply Gaussian process regression (see [17]), which can be viewed as obtaining an a posteriori distribution in a function space, conditioned on the given data. That is, we model the lifted states as a Gaussian process

$$z_t \sim \Psi(x_t) = \mathcal{GP}(\mu(x_t), k(x_t, x_t)). \tag{5}$$

Further, if we only have access to an output measurement y_t, which can be regarded as a function of the state x_t, the same lifting procedure can be applied to y_t, as this can be interpreted as indirectly lifting the state x_t. The measurements y_t can then be retrieved as a projection by the operator C from the lifted space of realized observables. This matrix C is also obtained through the subspace identification procedure, but in the Koopman interpretation, its rows contain the projection coefficients c from Eq. (4) corresponding to the particular observable functions associated with the output measurements y_t. In this interpretation a Koopman operator can be constructed that propagates the observables as functions of time without first specifying or learning a dictionary of basis functions.

The formulation can be extended to a control system on realizations of observables $z(t)$ with a control input at time t as $u_t \in \mathbb{R}^m$

$$z_{t+1} = Kz_t + Bu_t \tag{6a}$$

$$y_t = Cz_t + Du_t \tag{6b}$$

3. KOOPMAN OPERATOR VIA OBSERVABLES FROM MULTIPLE DATA RECORDS

We begin by collecting a dataset of input-output data from N trajectories of $n + 1$ timesteps each, $\mathcal{D}_i = \{u_t^i, y_t^i\}_{t=0}^n$ for $i = 1, \ldots, N$. We define the Hankel matrices for each trajectory of $n + 1$ measured output and input data and with l time delayed coordinates as,

$$Y_l^i = \begin{pmatrix} y_0^i & y_1^i & \cdots & y_{n-l}^i \\ y_1^i & y_2^i & \cdots & y_{n-l+1}^i \\ \vdots & \vdots & \ddots & \vdots \\ y_{l-1}^i & y_l^i & \cdots & y_n^i \end{pmatrix}, U_l^i = \begin{pmatrix} u_0^i & u_1^i & \cdots & u_{n-l}^i \\ u_1^i & u_2^i & \cdots & u_{n-l+1}^i \\ \vdots & \vdots & \ddots & \vdots \\ u_{l-1}^i & u_l^i & \cdots & u_n^i \end{pmatrix}.$$

In order to identify the system using the multiple trajectory records, these Hankel matrices are collected to form mosaic-Hankel matrices, see [18] as shown below

$$Y_l = \begin{bmatrix} Y_l^1 & Y_l^2 & \cdots & Y_l^N \end{bmatrix}, \qquad U_l = \begin{bmatrix} U_l^1 & U_l^2 & \cdots & U_l^N \end{bmatrix}, \quad (7)$$

where, $Y_l \in \mathbb{R}^{lp \times (n-l)N}$ and $U_l \in \mathbb{R}^{lm \times (n-l)N}$.

3.1 System matrix calculation using SSID

Assuming that we have collected the trajectory data and organized it into a mosaic Hankel matrix as mentioned above, we proceed by computing the lifted states Z_0^{lift} and system matrices K, C, B, and D using an SSID algorithm for multiple trajectories. We outline the procedure below, following closely the presentation of [14]. The system (6) can be concatenated and represented in the form as,

$$Y_l = \Gamma_l Z_0^{\text{lift}} + \mathcal{H}_l U_l \qquad (8)$$

where

$$\Gamma_l = \begin{bmatrix} C \\ CK \\ CK^2 \\ \vdots \\ CK^{l-1} \end{bmatrix}, \quad \mathcal{H}_l = \begin{bmatrix} D & 0 & 0 & \cdots & 0 \\ CB & D & 0 & \cdots & 0 \\ CKB & CB & D & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ CK^{l-2}B & \cdot & \cdot & \cdots & D \end{bmatrix}$$

$$Z_0^{\text{lift}} = \begin{bmatrix} z_0^1 & \cdots z_{n-l}^1 & z_0^2 & \cdots & \cdots & z_{n-l}^{N-1} & z_0^N \cdots z_{n-l}^N \end{bmatrix}$$

Where $\Gamma_l \in \mathbb{R}^{lp \times r}$ is the extended observability matrix, $\mathcal{H}_l \in \mathbb{R}^{lp \times lm}$ is the block-Toeplitz matrix and $Z_0^{\text{lift}} \in \mathbb{R}^{r \times (n-l)N}$ is the matrix of lifted states for the initial state of each column trajectory of mosaic Hankel matrix. The subspace system identification algorithm proceeds by projecting Eq. (8) onto the orthogonal complement of the row space of U_l. The projection operator for this is given by $\Pi_{U_l}^\perp = I - U_l^\mathsf{T}(U_l U_l)^\dagger U_l$ and this projection yields $Y_l \Pi_{U_l}^\perp = \Gamma_l Z_0^{\text{lift}} \Pi_{U_l}^\perp$. With this, we extract column space of Γ_l by taking the singular value decomposition of $Y_n \Pi_{U_n}^\perp$

$$Y_n \Pi_{U_n}^\perp = [Q_1 \, Q_2] \begin{bmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{bmatrix} \begin{bmatrix} V_1^\mathsf{T} \\ V_2^\mathsf{T} \end{bmatrix}, \qquad (9)$$

where Σ_1 contains the r largest singular values, and so we consider an approximation of Γ_l to be $\hat{\Gamma}_r = Q_1 \Sigma_1^{\frac{1}{2}}$. By selecting the r largest singular values we are determining the order of the lifted dynamics. Here, the first p rows of $\hat{\Gamma}_r$ give us the C matrix

$$C = \hat{\Gamma}_r(1:p,:), \qquad (10)$$

and with this, the K matrix can be found by solving the following least squares problem

$$K = [\hat{\Gamma}_r(1:(n-1)p,:)]^\dagger \, \hat{\Gamma}_r(p+1:np,:), \qquad (11)$$

where the : operator is used as in the familiar MATLAB syntax and $(\cdot)^\dagger$ is the Moore-Penrose pseudoinverse.

We again form a least squares problem to find matrices B, D and, Z_0^{lift}, using the equation below where $i = 1, 2, \ldots, N$ and $t = 0, 1, 2, \ldots, l$.

$$y_t^i = CK^t z_0^i + \sum_{k=1}^{t} CK^{(t-k)} B u_{k-1}^i + D u_t^i. \qquad (12)$$

We first vectorize and collect all outputs in one vector and write it in the form,

$$\begin{aligned} vec(Y) &= \begin{bmatrix} y_0^1 & \cdots & y_n^1 & y_0^2 & \cdots & \cdots y_n^{N-1} & y_0^N & \cdots & y_n^N \end{bmatrix}^\mathsf{T} \\ &= \begin{bmatrix} \Phi_B^\mathsf{T} & \Phi_D^\mathsf{T} & \Phi_{Z_0^{\text{lift}}}^\mathsf{T} \end{bmatrix} \begin{bmatrix} vec(B)^\mathsf{T} & vec(D)^\mathsf{T} & Z_0^{\text{lift}\mathsf{T}} \end{bmatrix}^\mathsf{T} \\ &= \Phi^\mathsf{T} \theta, \end{aligned}$$
$$(13)$$

With this, the matrices B, D and the initial lifted state Z_0^{lift} are given by the least square solution

$$\hat{\theta} = \begin{bmatrix} vec(B) \\ vec(D) \\ \hat{Z}_0^{\text{lift}} \end{bmatrix} = (\Phi \Phi^\mathsf{T})^{-1} \Phi \, vec(Y). \qquad (14)$$

4. GAUSSIAN PROCESSES AS BASIS FUNCTIONS

The SSID procedure outlined in the previous section for identifying the lifted state z and system matrices provides a method for obtaining the system matrices from input-output data collected from the system. However, it is also necessary to construct a mapping from the original states, x_t (or partial observation y_t) to the lifted state. For this, we employ Gaussian process regression, based on the idea discussed in Sec. 2 that the lifted states z_t obtained in subspace identification can be viewed as evaluations at a particular x_t of a dictionary function, which is itself a realization of a random variable in a function space. In Sec. 4.1 we give the procedure when measurements of the full state observable are available and in Sec. 4.2 we consider the case where only a partial observation is available.

4.1 Gaussian Processes as Basis functions of full state observable

The purpose of the Gaussian process model here is to give the mapping $z_t = \Psi(x_t)$ from the original system states $x \in \mathbb{R}^d$ to the lifted states $z \in \mathbb{R}^r$. For this, we use the data gathered above as Z_0^{lift} and corresponding states $X = [x_0^1, \cdots, x_{n-l}^1, \cdots, x_0^N, \cdots, x_{n-l}^N]$ obtained from the first p rows of the mosaic-Hankel matrix Y_l for the case when we have full state observable that is $y_t = x_t$. This data is then used to model each lifted state as a Gaussian process,

$$z_i(x)|\mathcal{D} \sim \mathcal{GP}(\mu_{z_i|\mathcal{D}}(x), k_{z_i|\mathcal{D}}(x,x)) \qquad (15)$$

where $i = 1, 2, \cdots, r$. The posterior mean and covariance are calculated as

$$\mu_{z_i|\mathcal{D}}(x) = \mu(x) + K_{xX}\left(K_{XX} + \sigma_n^2 I\right)^{-1} Z_0^{\text{lift}}(i,:)^\mathsf{T} \quad (16)$$

$$k_{z_i|\mathcal{D}}(x,x) = K_{xx} - K_{xX}\left(K_{XX} + \sigma_n^2 I\right)^{-1} K_{Xx} \quad (17)$$

where the kernel matrices are found by evaluating the kernel function. In particular, we are using the ARD squared exponential kernel. The kernel parameters can be optimized through log-likelihood maximization. Here this is done using the MATLAB toolbox for Gaussian process regression to fit each of the r GPs independently.

4.2 Gaussian Processes as Basis functions of time delayed observables

Often in system identification theory [9], using time-delayed state observables to identify the whole nonlinear states can be an effective way to estimate the state of a system even if the full state is not directly measured. Therefore, when the output is not a full state, the Gaussian process model gives the mapping from the output and the time-delayed coordinates of outputs and inputs to the lifted states $z_t = \Psi(\hat{x})$.

$$\hat{x}_t = [y_t, y_{t-1}, \cdots, y_{t-k}, u_{t-1}, \cdots, u_{t-k}]^\mathsf{T}$$
$$\hat{X} = [\hat{x}_0^1, \cdots, \hat{x}_{n-l}^1, \cdots, \hat{x}_0^N, \cdots, \hat{x}_{n-l}^N]. \quad (18)$$

These time-delayed observables $\hat{X}$ with the corresponding realizations of the lifted state Z_0^{lift} is used to model the lifted state as Gaussian process.

$$z_i(\hat{x})|\mathcal{D} \sim \mathcal{GP}(\mu_{z_i|\mathcal{D}}(\hat{x}), k_{z_i|\mathcal{D}}(\hat{x}, \hat{x})). \quad (19)$$

The posterior mean and covariance are calculated using (16) and (17).

5. PREDICTION USING GP-SSID

For predicting the evolution of the system from a given initial condition x_0, its corresponding initial condition of the lifted state is represented as a Gaussian random variable, found by evaluating the GP at x_0.

$$z_0 \sim \mathcal{GP}(\mu_{z|\mathcal{D}}(x_0), k_{z|\mathcal{D}}(x_0, x_0)) \quad (20)$$

The mean and covariance of the initial lifted state are found by evaluating the posterior mean and covariance for each lifting function, using (16) and (17).

$$\mu_{z|\mathcal{D}}(x_0) = \begin{bmatrix} \mu_{z_1|\mathcal{D}}(x_0) & \cdots & \mu_{z_r|D}(x_0) \end{bmatrix}^\mathsf{T} \quad (21a)$$

$$k_{z|\mathcal{D}}(x_0, x_0) = \text{diag}\begin{bmatrix} k_{z_1|\mathcal{D}}(x_0, x_0) & \cdots & k_{z_r|D}(x_0, x_0) \end{bmatrix} \quad (21b)$$

Then, denote the mean and covariance of the lifted state by $\bar{z} = \mu_{z|\mathcal{D}}$ and $P = k_{z|\mathcal{D}}$, respectively. This mean and covariance are propagated forward in time using the system matrices as

$$\bar{z}_{t+1} = K\bar{z}_t + Bu_t \quad (22a)$$
$$P_{t+1} = KP_t K^\mathsf{T} \quad (22b)$$

and the estimated mean of the output, $\hat{y}$ along with its covariance Q at each step can be obtained as

$$\hat{y}_t = C\hat{z}_t + Du_t \quad (23a)$$
$$Q_t = C\, P_t\, C^\mathsf{T} \quad (23b)$$

where t is the time index.

6. RESULTS

6.1 Linear predictor without Control

We apply the proposed method to obtain the Koopman operator for the nonlinear Duffing oscillator using multiple and single data records. Then, we compare the open-loop prediction of each linear model against the true trajectory simulated using the nonlinear model,

$$\dot{x}_1 = x_2$$
$$\dot{x}_2 = 4x_1 - x_1^3 \quad (24)$$
$$y = \begin{bmatrix} x_1 & x_2 \end{bmatrix}^\mathsf{T}$$

The training data set is collected by discretizing the nonlinear system (24) using a fourth order Runge-Kutta method and time step of 0.01s. For the first linear model (multiple data records), we sample 200 trajectories all randomly initialized for $x \in \mathbb{R}^2$ in the range $[-3, 3]$ and simulate each trajectory for 150-time steps. We then con-
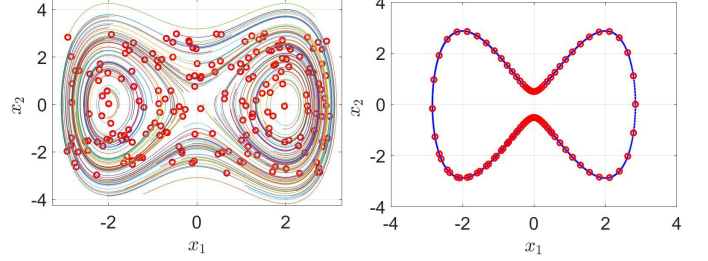

Fig. 2. (a) training data set for the model with multiple data records (b) training data for the single trajectory case. Red points are the training data for the GP and multicolored lines show the full trajectories.

struct the mosaic-Hankel matrix and follow the procedure discussed in section 3 to construct a higher dimensional linear system. For the second model, we sample a single trajectory of 8000 time steps long to construct the Koopman operator. We select 200 data points of the system states and the corresponding time realization of the basis functions (Z_{lift}) from both models, which we use to train the Gaussian processes. In the case of the multiple trajectory model, we select $r = 10$, while for the single trajectory model, we select $r = 20$. For predicting the forward dynamics, we first use the respective trained Gaussian processes to lift the initial states to the lifted space and then propagate forward using equations (22) and (23).
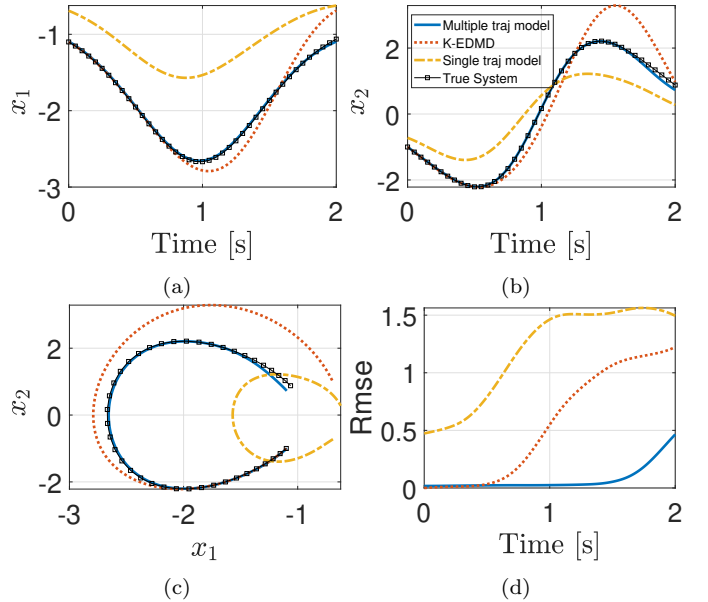

Fig. 3. Comparison of the prediction of multiple trajectory model (solid blue line), single trajectory model (Dash-dot yellow line) and Koopman EDMD model (dotted red line) with the solution obtained by direct integration of the nonlinear system (dash-square black line) (24) for the initial state [-1.1,-1].

To compare the prediction of the proposed GP-SSID algorithm to the traditional EDMD algorithm, see for instance [5], we use the same data from the multiple trajectory data set to construct the Koopman operator using the EDMD method with the choice of 25 radial basis functions and the states themselves, which gives us the total number of basis functions $r = 27$. We then compare predictions using 500 randomly sampled initial points in the state space, enabling us to predict a 2-second trajectory for each initial point. The average root mean squared error for the SSID model trained with multiple trajectories is 0.0728, while for the single trajectory model, it is 1.2093, and for the K-EDMD method average RMSE is 0.5424. Figure 3(a-c) shows one of the prediction example with an initial state at $[-1.1, -1]$. Figure 3d illustrates the increase in RMSE as the prediction time increases.

6.2 MPC with Koopman operator

In this section, we consider a Duffing oscillator with a control input

$$
\begin{aligned}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= -0.2x_2 + 4x_1 - x_1^3 + u \\
y &= [x_1 \quad x_2]^\mathsf{T}
\end{aligned} \tag{25}
$$

For comparison purposes, we identify two linear models, one based on multiple data records and the other on a single trajectory. Next we use linear model predictive control (MPC) technique to control the nonlinear system (25) and compare these two linear models in terms of the performance index of the MPC and convergence to the desired output.

We gather training data sets shown in figure 4; for the first model, we use multiple data records by simulating 100 randomly sampled initial states forward for 500-time steps with random control inputs $u \in [-3, 3]$. For the second model, we propagate an initial condition for 8000-time steps with a random control input of a similar range. Subsequently, we construct mosaic-Hankel matrices from the collected data, identify the linear system matrices, and train the basis function (Gaussian Processes) for both models, as discussed in sections 3 and 4. The size of the lifted states selected is $r = 35$ for both the models.
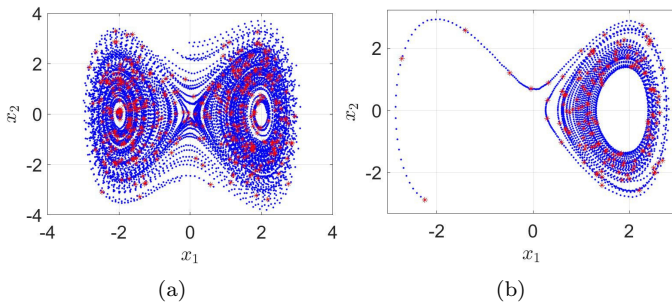

Fig. 4. Training data set for the Duffing oscillator (25). Red points are the training data for the GPs, and the blue points show the full trajectories. (a) multiple data records, (b) single trajectory

Linear model predictive control (MPC) techniques is utilized with the identified linear models to guide the system's states toward a desired state/trajectory. MPC optimizes the performance of dynamic systems by solving a sequence of optimization problems, determining the best control inputs that minimize the deviation between the desired and actual outputs while ensuring compliance with imposed constraints. We can formulate the MPC as a quadratic optimization problem with constraints as

$$
\min_u \sum_{j=t_i}^{t_i+N_p} \frac{1}{2}\Big((y_j - y_j^r)^\mathsf{T} Q_j (y_j - y_j^r) + u_j^\mathsf{T} R_j u_j \Big) \tag{26}
$$

$$
\begin{aligned}
\text{s.t.} \quad & Z_0 = \mathcal{GP}(\mu(x_0), k(x_0, x_0)) \\
& Z_{j+1} = KZ_j + Bu_j, \qquad j = t_i. \ldots . N_p \\
& y_j = CZ_j \\
& u_{min} \le u_j \le u_{max}
\end{aligned}
$$

Where $Q_j = 5$, $Q_{N_p} = 10$ and $R_j = 10^{-3}$ are the positive semi-definite weight matrices for the outputs and inputs, t_i is the current time, $N_p = 100$ is the prediction horizon and y^r is the reference trajectory that is to be tracked. In this example, we want to drive a given initial state x_0
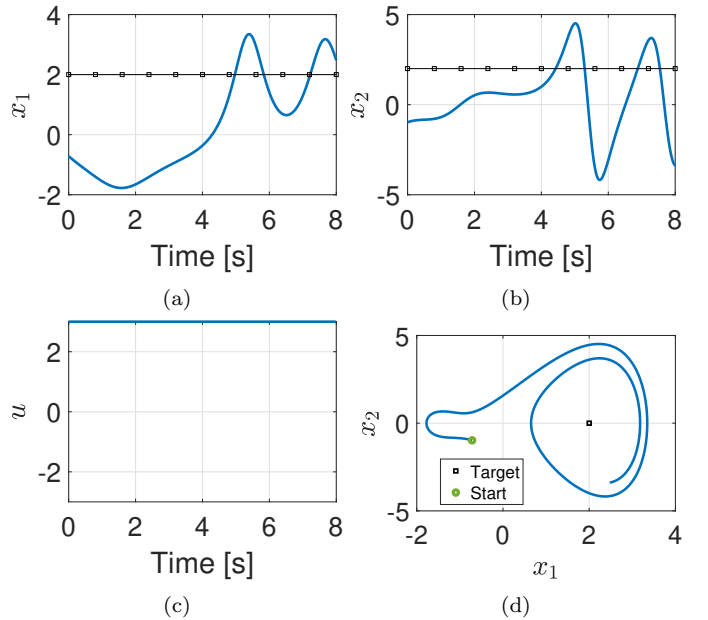

Fig. 5. MPC results for linear model constructed using single trajectory data. The initial state $x_0 = [-0.7160, -0.9789]$ and the target state is $[2, 0]$.

to a desired state $x_{des} = [2, 0]$. At the start of the MPC, the Gaussian processes are used to lift the initial state x_0. Then MPC predicts for N_p-time step, and one control step is implemented, which updates the initial state. Each of these feedback updates are passed again through the Gaussian process, and the process continues. Figure 5 shows the results for the model which uses single trajectory data, and in figure 6 are the results for the model using multiple data records. From both figures, we can observe that the multiple data record model performs better and tracks the desired reference trajectory, as seen in figure 6.

To assess the effectiveness of our proposed methodology in contrast to established data-driven techniques for addressing the presented optimal control problem, we leverage the K-EDMD algorithm as outlined in [5]. We employ this algorithm to construct a linear system using the same multiple trajectory dataset. This construction involves using
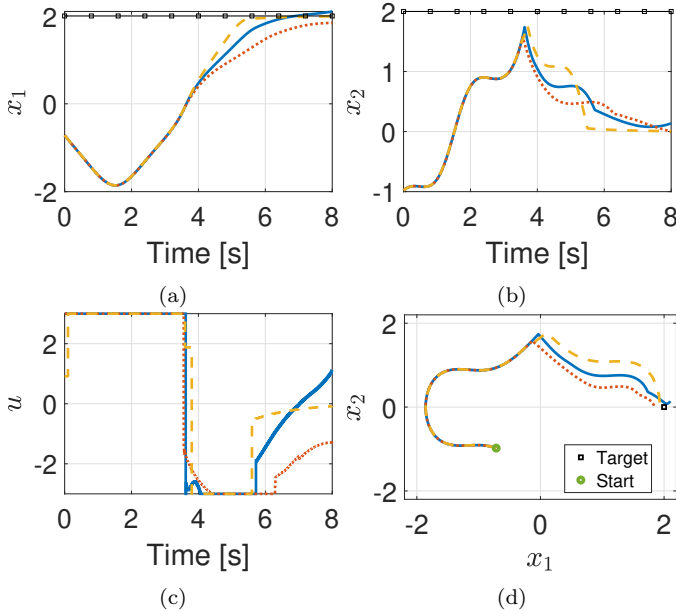
Fig. 6. Comparison of the MPC results from the linear models constructed using GP-SSID with multiple data records (solid blue line) and K-EDMD (dotted red line) with model-based nonlinear MPC solutions (dash-dot yellow line). The initial state $x_0 = [-0.7160, -0.9789]$ and the target state is $[2, 0]$.

36 radial basis functions alongside the states themselves, leading to generating a Koopman operator with a total of $r = 38$ basis functions. Subsequently, we apply a Model Predictive Control (MPC) formulation, as detailed in Section 6.2, tailored to the linear system derived through the K-EDMD algorithm. The results of our MPC solutions are illustrated in figure 6. Notably, our proposed methodology exhibits faster convergence compared to the K-EDMD approach. Furthermore, the terminal error is larger in the case of the K-EDMD algorithm.

While the GP-SSID based Koopman linear MPC is well suited for data-driven systems, we also compare this approach with a purely model-based nonlinear model predictive control (NMPC) approach with a similar cost function. We observe that the model-based controller exhibits faster convergence and achieves a lower cost when compared to the data-driven Koopman operator-based linear controllers. However, the computational time required for linear MPC based on both the GP-SSID and K-EDMD approaches is lower in comparison to the computational demands of the nonlinear model-based controller for the same time step $\delta t = 0.01s$. As an illustration of the difference in computational time, for the example shown in fig. 6, GP-SSID and K-EDMD methods take about 101s and 105s, respectively, and the nonlinear MPC takes 1108s.

The GP-SSID method consistently outperforms the K-EDMD algorithm in terms of prediction accuracy, as illustrated in figure 3 and while it seems that the MPC implementations of both approaches (as shown in fig. 6) yields similar solutions, this similarity in performance is a result of using a one-step control horizon, continually updating the initial conditions. Nonetheless, it's essential to note that the one-step control horizon also entails a higher computational cost and may make online implementation

infeasible. Therefore we evaluate the performance of both algorithms with a larger control horizon of $N_c = 50$. For the example considered in fig. 6 when N_c is increased to 50 the computational time for both GP-SSID and K-EDMD approaches is about 1.7s although the terminal error with the K-EDMD approach grows compared to the GP-SSID approach.
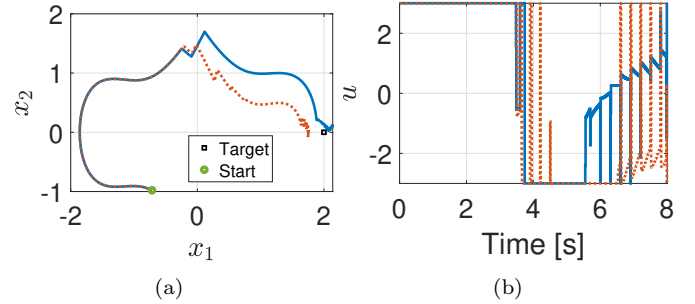

Fig. 7. Comparison of MPC results with control horizon $N_c = 50$ obtained from the linear models constructed using GP-SSID with multiple data records (in solid blue line) and K-EDMD (dotted red line). The initial state $x_0 = [-0.7160, -0.9789]$ and the target state is $[2, 0]$.

6.3 MPC with Koopman operator for partially observed state

Consider the Duffing oscillator with control in (25), which is partially observed such that

$$y = x_2 \qquad (27)$$

Collecting data and identifying the Koopman operator using subspace identification remains similar to the previous example in section 6.2. We sample 100 trajectories with random initial states and propagate them for 400 time steps with random control inputs. To train the basis function (Gaussian processes), we use output y along with 5 time-delayed inputs and outputs with the corresponding time realizations of the lifted function $Z_{\text{lift}} \in \mathbb{R}^{25}$. We formulate a similar MPC model (26) as discussed in the previous section to control the output x_2 to track reference signal. Here, the weight matrices used are $Q_j = 100$ and $R_j = 0.001$. The results in figure 8 show that we can use this method to track a reference trajectory for a partially observed system.
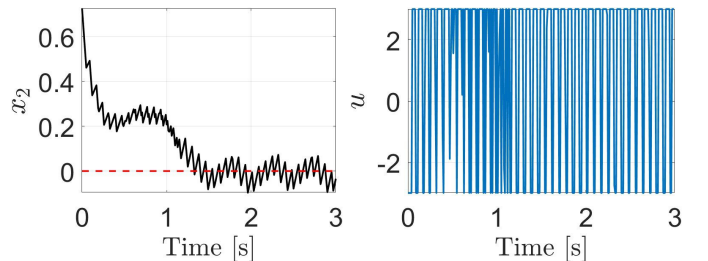

Fig. 8. MPC result for tracking the observable x_2 to 0.

7. CONCLUSION

In this paper, we have explored connections between the classical system identification technique of subspace

identification and recently developed methods for system identification based on constructing a finite-dimensional approximation of the Koopman operator. In particular, we have shown that these methods can be directly related by considering the latent state identified through SSID to be an evaluation of a set of dictionary functions at a particular state of the underlying nonlinear system. With this formulation, we have shown that for nonlinear systems, it is necessary to consider multiple data records, as a single trajectory does not sufficiently explore the state space in most cases. This is indicated by improved prediction and control performance. We also show that the same formulation can be applied in cases where only partial state measurements are available.

REFERENCES

[1] Rowley, C. W., Mezic, I., Bagheri, S., Schlatter, P., and Henningson, D. S., 2009, "Spectral analysis of nonlinear flows," *Journal of Fluid Mechanics,* **641**, p. 115–127.

[2] Susuki, Y., and Mezić, I., 2014, "Nonlinear koopman modes and power system stability assessment without models," *IEEE Transactions on Power Systems,* **29**(2), pp. 899–907.

[3] Hou, Z.-S., and Wang, Z., 2013, "From model-based control to data-driven control: Survey, classification and perspective," *Information Sciences,* **235**, pp. 3–35.

[4] Lasota, A., and Mackey, M. C., 1994, *Chaos, Fractals, and Noise : Stochastic Aspects of Dynamics* Springer.

[5] Williams, M. O., Kevrekidis, I. G., and Rowley, C. W., 2015, "A data–driven approximation of the Koopman operator: Extending dynamic mode decomposition," *Journal of Nonlinear Science,* **25**(6), pp. 1307–1346.

[6] Klus, S., Koltai, P., and Schütte, C., 2015, "On the numerical approximation of the Perron-Frobenius and Koopman operator," *arXiv preprint arXiv:1512.05997.*

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

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

[9] Ljung, L., 1998, *System identification* Springer.

[10] Overschee, P., and De Moor, B., 1996, *Subspace Identification for Linear Systems* Springer.

[11] Lian, Y., and Jones, C. N., 2019, "Learning feature maps of the koopman operator: A subspace viewpoint," In 2019 IEEE 58th Conference on Decision and Control (CDC), pp. 860–866.

[12] Lian, Y., and Jones, C. N., 2020, "On gaussian process based koopman operators," *IFAC-PapersOnLine,* **53**(2), pp. 449–455.

[13] Willems, J. C., Rapisarda, P., Markovsky, I., and De Moor, B. L., 2005, "A note on persistency of excitation," *Systems & Control Letters,* **54**(4), pp. 325–329.

[14] Holcomb, C. M., and Bitmead, R. R., 2017, "Subspace identification with multiple data records: unlocking the archive," *arXiv preprint arXiv:1704.02635.*

[15] Li, Q., Deitrich, F., Bollt, E. M., and Kevrekidis, I. G., 2017, "Extended dynamic mode decomposition with dictionary learning: A data-driven adaptive spectral decomposition of the Koopman operator," *CHAOS,* **27**, p. 103111.

[16] Lusch, B., Kutz, J. N., and Brunton, S. L., 2018, "Deep learning for universal linear embeddings of nonlinear dynamics," *Nature communications,* **9**(1), p. 4950.

[17] Rasmussen, C. E., and Williams, C. K. I., 2006, *Gaussian processes for machine learning* Adaptive computation and machine learning. MIT Press, Cambridge, Mass OCLC: ocm61285753.

[18] van Waarde, H. J., De Persis, C., Camlibel, M. K., and Tesi, P., 2020, "Willems' Fundamental Lemma for State-Space Systems and Its Extension to Multiple Datasets," *IEEE Control Systems Letters,* **4**(3), July, pp. 602–607.