Dynamic Mode Decomposition with Gaussian Process Regression for Control of High-Dimensional Nonlinear Systems

Alexandros Tsolovikos[*] **Efstathios Bakolas, and David Goldstein**
Department of Aerospace Engineering and Engineering Mechanics
The University of Texas at Austin
Austin, TX 78712

In this work, we consider the problem of learning a blackuced-order model of a high-dimensional stochastic nonlinear system with control inputs from noisy data. In particular, we develop a hybrid parametric/non-parametric model that learns the "average" linear dynamics in the data using dynamic mode decomposition with control (DMDc) and the nonlinearities and model uncertainties using Gaussian process (GP) regression and compare it with total least squares dynamic mode decomposition, extended here to systems with control inputs (tlsDMDc). The proposed approach is also compablack with existing methods, such as DMDc-only and GP-only models, in two tasks: controlling the stochastic nonlinear Stuart-Landau equation and pblackicting the flowfield induced by a jet-like body force field in a turbulent boundary layer using data from large-scale numerical simulations.

1 Introduction

Control of high-dimensional nonlinear dynamical systems, such as turbulent flows, which are expensive to model due to their large state spaces, has led to the need for approximate models that are computationally tractable and amenable to standard control algorithms. Simulations and experiments of such complex and uncertain dynamical systems typically yield large spatiotemporal data that can be used to learn tractable blackuced-order models suitable for control.

Data-driven model blackuction of systems with high-dimensional state spaces has typically been performed using proper orthogonal decomposition (POD) [1]. POD computes the singular value decomposition (SVD) of snapshots of the high-dimensional state (e.g. flowfield) and identifies an optimal subspace containing the most energetic modes on which the high-dimensional state space is projected [2]. For systems with control inputs, POD variants, such as balanced proper orthogonal decomposition (BPOD) [3], have also been proposed, although BPOD requires knowledge of the underlying equations of the system. Dynamic mode decomposition (DMD) [4, 5] is an attractive alternative for modeling the spatiotemporal evolution of data, such as fluid flows, from time-resolved snapshots of the high-dimensional system. Extensions of DMD have included systems with control inputs [6, 7], noise-aware [8, 9], and sparsity-promoting variants [10], among others.

DMD-based approaches often fail to capture the nonlinearities in the underlying dynamics due to the assumption of linearity of the DMD models. For that reason, Koopman operator theory – which is viewed as a generalization of DMD – has been extensively exploblack both in the fluid dynamics [11] and the controls community [12, 13], with DMD-based [14, 15], kernel-based [16], and deep learning-based [17, 18] approximations of the Koopman operator proposed. Similar modeling efforts have also led to new methods such as sparse identification of nonlinear dynamics (SINDy) [19] and operator inference for model blackuction [20].

Gaussian Process regression [21] is a type of non-parametric regression model describing distributions over functions conditioned on the training data. They are ideal for learning arbitrary nonlinear stochastic dynamics due to their flexibility and inherent ability to provide uncertainty estimates capturing both process and measurement noise, as well as model uncertainties. GP regression and its scalable variants [22, 23, 24, 25] have been successfully used for a number of dynamical systems modeling [26, 27] and control tasks [28, 29, 30]. In the context of high-dimensional systems, GP regression has been used in modeling the POD coefficients for systems with varying parameters [31, **?**, 32, 33] as well as the blackuced-order dynamics after POD or autoencoder-based order blackuction [34, 35].

In this brief, we propose a hybrid DMDc + GP model for learning the nonlinearities and model uncertainties in the blackuced-order data that DMDc alone fails to capture. In addition, total least squares DMD [9] is extended to systems with control inputs (tlsDMDc) to fairly compare the proposed method with a noise-aware, linear-only DMD-based approach that accounts for control inputs.

In Section 2, we first introduce DMDc and extend total least-squares DMD [9] to systems with control inputs. Then, we introduce the general framework of Gaussian process regression and the process of training a DMDc + GP blackuced-order model. In Section 3, we demonstrate the

[*]Corresponding author. Email: tsolovikos@utexas.edu

advantages of the proposed models on two tasks: modeling and controlling the nonlinear stochastic Stuart-Landau equation from noisy data and pblackicting the wall-normal velocity field induced by a jet-like body force field in a turbulent boundary layer.

2 Method

2.1 Dynamic Mode Decomposition with Control

Assume that the unknown stochastic nonlinear and, possibly, high-dimensional system we want to model has discrete-time dynamics of the form

$$\mathbf{y}(k+1) = \mathbf{f}(\mathbf{y}(k), \mathbf{u}(k), \mathbf{w}(k)), \tag{1}$$

where $\mathbf{y} \in \mathbb{R}^{n_y}$ is the state, $\mathbf{u} \in \mathbb{R}^{n_u}$ the control input, $\mathbf{w} \in \mathbb{R}^{n_y}$ the independent and identically distributed Gaussian white noise, and $\mathbf{f}(\cdot)$ the nonlinear operator that propagates the state $\mathbf{y}(k)$ by one time step.

Our goal is to identify a blackuced-order model of the underlying dynamics when (1) is unknown and only a limited number of noisy experimental or numerical data is available, as is typical in fluid dynamics applications.

In particular, given a set of p training data tuples $\{(\mathbf{y}^{(i)}, \mathbf{u}^{(i)}, \mathbf{f}(\mathbf{y}^{(i)}, \mathbf{u}^{(i)}, \mathbf{w}^{(i)}))\}_1^p$, where $\mathbf{y}^{(i)}$, $i = 1, \ldots, p$, are not necessarily sequential, the data can be arranged as

$$\mathbf{Y} = \begin{bmatrix}\mathbf{y}^{(1)} \cdots \mathbf{y}^{(p)}\end{bmatrix} \in \mathbb{R}^{n_y \times p}, \tag{2a}$$

$$\mathbf{Y}' = \begin{bmatrix}\mathbf{f}(\mathbf{y}^{(1)}, \mathbf{u}^{(1)}, \mathbf{w}^{(1)}) \cdots \mathbf{f}(\mathbf{y}^{(p)}, \mathbf{u}^{(p)}, \mathbf{w}^{(p)})\end{bmatrix} \in \mathbb{R}^{n_y \times p}, \tag{2b}$$

$$\mathbf{U} = \begin{bmatrix}\mathbf{u}^{(1)} \cdots \mathbf{u}^{(p)}\end{bmatrix} \in \mathbb{R}^{n_u \times p}. \tag{2c}$$

2.1.1 Model Order blackuction

A common way to blackuce the dimensionality of the data when $n_y \gg 1$ is to project the high-dimensional state $\mathbf{y}(k)$ onto the POD modes given by the SVD of the data matrix $\mathbf{Y} = U\Sigma V^{\mathrm{T}}$, where the columns of matrix $U \in \mathbb{R}^{n_y \times p}$ are the orthonormal eigenvectors of $\mathbf{Y}\mathbf{Y}^{\mathrm{T}}$ or POD modes arranged by their energy content, i.e. their singular value, the columns of $V \in \mathbb{R}^{p \times p}$ are the orthonormal eigenvectors of $\mathbf{Y}^{\mathrm{T}}\mathbf{Y}$, and $\Sigma \in \mathbb{R}^{p \times p}$ is the diagonal matrix containing the singular values of $\mathbf{Y}$ arranged by their magnitude.

Projecting the high-dimensional snapshots $\mathbf{y}(k)$ on the range space of the matrix $U_{POD} \in \mathbb{R}^{n_y \times n_x}$ formed by the first (most energetic) n_x POD modes corresponding to the n_x largest singular values of $\mathbf{Y}$ is a common choice in model blackuction methods that focuses the modeling efforts on the most important modes of the high-dimensional system and ignores the least energetic (and, typically, noisy) ones. The high-dimensional state can then be approximated as

$$\mathbf{y}(k) \approx U_{POD}\mathbf{x}(k), \tag{3}$$

where $\mathbf{x}(k) \in \mathbb{R}^{n_x}$ is the amplitude vector of the POD modes at time step k. In general, $\mathbf{x}(k)$ is approximated in the least-squares sense as $\mathbf{x}(k) = U_{POD}^{\mathrm{T}}\mathbf{y}(k)$. The snapshot matrices (2a) - (2b) are also blackuced as

$$\mathbf{X} = U_{POD}^{\mathrm{T}}\mathbf{Y}, \quad \mathbf{X}' = U_{POD}^{\mathrm{T}}\mathbf{Y}',$$

where $\mathbf{X}, \mathbf{X}'$ are the POD mode amplitude matrices for the training data (2a) and (2b).

2.1.2 Dynamic Mode Decomposition with Control

In order to capture the linear part of the controlled dynamical system, the POD mode amplitudes $x(k)$ are modeled as a discrete-time linear state space system of the form

$$\mathbf{x}(k+1) = A\mathbf{x}(k) + B\mathbf{u}(k) + \mathbf{e}(x(k), u(k), w(k)), \tag{4}$$

where $A \in \mathbb{R}^{n_x \times n_x}$ and $B \in \mathbb{R}^{n_x \times n_u}$ are the state and control transition matrices and $\mathbf{e}(\cdot)$ is the linearization error from the unmodeled nonlinear and process noise part of the dynamics. The linear part of the dynamics can be computed by solving the following least-squares minimization problem:

$$\min_{A,B} \quad \|\mathbf{X}' - A\mathbf{X} - B\mathbf{U}\|_F^2 \tag{5}$$

whose minimizer is given by

$$\begin{bmatrix}A \; B\end{bmatrix} = \mathbf{X}'\begin{bmatrix}\mathbf{X}\\ \mathbf{U}\end{bmatrix}^{\dagger}, \tag{6}$$

where $\dagger$ denotes the Moore-Penrose inverse. Now, the linear part of the dynamics can be modeled by setting $\mathbf{e}(\cdot) = 0$ in (4), which is the approach that DMDc follows [6]. The full-dimensional snapshot $\mathbf{y}(k)$ at time step k can then be approximated from (3).

2.1.3 Total Least Squares Dynamic Mode Decomposition with Control

In [9], total least squares dynamic mode decomposition (tlsDMD) was proposed in order to account for the presence of measurement and process noise in the data. Following [9], we start by assuming that the snapshots (2a) and (2b) can be decomposed in a mean and noise part as

$$\mathbf{Y} = \bar{\mathbf{Y}} + \mathbf{E}_Y, \quad \mathbf{Y}' = \bar{\mathbf{Y}}' + \mathbf{E}_{Y'}$$

where $\bar{\mathbf{Y}}, \bar{\mathbf{Y}}'$ are the mean snapshots and $\mathbf{E}_Y, \mathbf{E}_{Y'}$ are the noise and modeling errors $\mathbf{e}(\cdot)$ stacked similarly to (2a) and (2b). After projection on the POD modes, we get

$$\mathbf{X} = \bar{\mathbf{X}} + \mathbf{E}_X, \quad \mathbf{X}' = \bar{\mathbf{X}}' + \mathbf{E}_{X'}.$$

According to [9], the least-squares minimization approach of (5) in DMD (and, consequently, DMDc) accounts only for the noise $\mathbf{E}_{X'}$ in the plus-one time step data $\mathbf{X}'$, leading to a bias in the estimate of the dynamics that depends on $\mathbf{E}_X$. Alternatively, one can use total least-squares DMD to account for noise in both matrices $\mathbf{X}$ and $\mathbf{X}'$. The approximation of the dynamics can be expressed as

$$\bar{\mathbf{X}}' + \mathbf{E}_{X'} = A(\bar{\mathbf{X}} + \mathbf{E}_X) + B\mathbf{U} \tag{7}$$

and the error in both components can be minimized simultaneously by solving the least-squares minimization problem

$$\min_{A,B} \quad \left\| \begin{bmatrix} \mathbf{E}_X \ \mathbf{E}_{X'} \end{bmatrix} \right\|_F^2 . \tag{8}$$

Equation (7) can be reformulated as

$$\begin{bmatrix} A \ B \ -I \end{bmatrix} \begin{bmatrix} \bar{\mathbf{X}} + \mathbf{E}_X \\ \mathbf{U} \\ \bar{\mathbf{X}}' + \mathbf{E}_X' \end{bmatrix} = 0 \tag{9}$$

and the solution to (8) can be computed using the truncated SVD

$$\begin{bmatrix} \bar{\mathbf{X}} + \mathbf{E}_X \\ \mathbf{U} \\ \bar{\mathbf{X}}' + \mathbf{E}_X' \end{bmatrix} = U\Sigma_{n_x+n_u}V^* = \begin{bmatrix} U_{11} \ U_{12} \\ U_{21} \ U_{22} \end{bmatrix} \begin{bmatrix} \Sigma_1 \ 0 \\ 0 \ \ 0 \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \end{bmatrix}$$

where only the first $n_x + n_u$ singular values are kept, leading to an unbiased estimate of A and B

$$\begin{bmatrix} A \ B \end{bmatrix} = U_{21}U_{11}^{-1}. \tag{10}$$

The above is an extension of tlsDMD to systems with control inputs (tlsDMDc).

2.2 Gaussian Process Regression

Once we model the linear dynamics of the unknown system with DMDc or tlsDMDc and we obtain matrices A and B, we can then learn the nonlinear and process noise terms $e(\cdot)$ of the dynamics (4) that DMDc (and tlsDMDc) fails to capture. With A and B known, we can estimate the linear approximation error of each snapshot as

$$\boldsymbol{\varepsilon} = \mathbf{X}' - A\mathbf{X} - B\mathbf{U}. \tag{11}$$

While DMDc and tlsDMDc assumes that the error term $e(\cdot)$ in (4) is zero, here we attempt to model the error with GP regression trained on the data in (11). In particular, we seek to model the error term $\mathbf{e}(\mathbf{x}(k), \mathbf{u}(k), \mathbf{w}(k))$ in (4) with a non-parametric Gaussian process regression model as follows.

2.2.1 Exact GP Inference

We start by considering a single component of the vector function $\mathbf{e}(\cdot)$ which, for clarity, we will refer to as $e(\cdot)$. The input to this function at time step k is the concatenation of the state $\mathbf{x}(k)$ and control input $\mathbf{u}(k)$ at that time step, i.e. $\mathbf{z}(k) = \begin{bmatrix} \mathbf{x}^{\mathrm{T}}(k) \ \mathbf{u}^{\mathrm{T}}(k) \end{bmatrix}^{\mathrm{T}} \in \mathbb{R}^{n_z}$, with $n_z = n_x + n_y$. If we have noisy observations ε_i of the unknown scalar-valued function $e(\cdot) : \mathbb{R}^{n_z} \to \mathbb{R}$ at known inputs $\mathbf{z}_i$, for all $\mathbf{Z} = \{\mathbf{z}_i\}_{i=1}^{p}$, we collect these observations in a vector ε.

Let the measurement likelihood $p(\varepsilon_i \mid e(\mathbf{z}_i))$ be zero-mean Gaussian and let $\mathbf{e}$ be the (unknown) vector containing the values of $e(\cdot)$ at the points $\mathbf{Z}$. We introduce a Gaussian prior $e(\mathbf{z}) \sim \mathcal{N}(e(\mathbf{z}) \mid m(\mathbf{z}), k(\mathbf{z},\mathbf{z}))$, where $m(\cdot) : \mathbb{R}^n \to \mathbb{R}$ is the *mean function* (typically chosen to be the zero function) and $k(\cdot,\cdot) : \mathbb{R}^{n_z} \times \mathbb{R}^{n_z} \to \mathbb{R}$ is the *kernel function* (typically, a squablack exponential) that measures the closeness between two input points and specifies the smoothness and continuity properties of the underlying unknown function $e(\cdot)$.

The prior over the entire vector $\mathbf{e}$ can now be written as

$$p(\mathbf{e} \mid \mathbf{Z}) = \mathcal{N}(\mathbf{e} \mid m(\mathbf{Z}), k(\mathbf{Z},\mathbf{Z})), \tag{12}$$

with the mean vector defined as $[m(\mathbf{Z})]_i = m(\mathbf{z}_i)$ and the covariance $[k(\mathbf{Z},\mathbf{Z})]_{ij} = k(\mathbf{z}_i, \mathbf{z}_j)$.

The joint density of vectors ε (known) and $\mathbf{e}$ (unknown) is

$$p(\varepsilon, \mathbf{e} \mid \mathbf{Z}) = p(\varepsilon \mid \mathbf{e}, \mathbf{Z}) p(\mathbf{e} \mid \mathbf{Z}). \tag{13}$$

With Gaussian likelihood $p(\varepsilon \mid \mathbf{e}, \mathbf{Z}) = \mathcal{N}\left(\varepsilon \mid \mathbf{e}, \sigma_\varepsilon^2 I\right)$, the marginal likelihood

$$\begin{aligned} p(\varepsilon|\mathbf{Z}) &= \int p(\varepsilon \mid \mathbf{e}, \mathbf{Z}) p(\mathbf{e} \mid \mathbf{Z}) d\mathbf{e} \\ &= \mathcal{N}\left(\varepsilon \mid m(\mathbf{Z}), k(\mathbf{Z},\mathbf{Z}) + \sigma_\varepsilon^2 I\right) \end{aligned} \tag{14}$$

is analytically computed and the *hyperparameters* $\Theta = \{\theta_m, \theta_k, \sigma_\varepsilon\}$ that define the Gaussian process mean, kernel, and likelihood functions can be found by minimizing the negative log-likelihood of the training data,

$$\Theta_{\text{opt}} = \arg\min_{\Theta} (-\log p(\varepsilon \mid \mathbf{Z})). \tag{15}$$

Pblackiction of ε_* on a new state-input pair $\mathbf{z}_*$ is done by conditioning on the training data,

$$p(\varepsilon_* \mid \mathbf{z}_*, \varepsilon, \mathbf{Z}) = \int p(\varepsilon_*, \varepsilon \mid \mathbf{z}_*, \mathbf{Z}) d\varepsilon = \mathcal{N}(\varepsilon_* \mid \mu_*, \sigma_*),$$

where

$$\begin{aligned} \mu_* &= m(\mathbf{z}_*) + k(\mathbf{z}_*, \mathbf{Z}) \left[k(\mathbf{Z},\mathbf{Z}) + \sigma_\varepsilon^2 I\right]^{-1} (\varepsilon - m(\mathbf{Z})), \\ \sigma_* &= k(\mathbf{z}_*, \mathbf{z}_*) - k(\mathbf{z}_*, \mathbf{Z}) \left[k(\mathbf{Z},\mathbf{Z}) + \sigma_\varepsilon^2 I\right]^{-1} k(\mathbf{Z}, \mathbf{z}_*). \end{aligned}$$

2.2.2 Multiple Outputs

So far, the error $e_i \in \mathbb{R}$ we have consideblack has been a scalar. In the general case, the error in (4) will be a vector

$\mathbf{e}_i \in \mathbb{R}^{n_x}$. We can train the hyperparameters of an exact GP using the estimated error snapshots $\mathcal{E}$ in (11) and defining the matrix $\mathbf{E}$ as the matrix containing the function values $\mathbf{e}(\mathbf{z}_i)$ at the i-th row. The latent functions are now $e_d(\cdot) : \mathbb{R}^{n_z} \to \mathbb{R}$, for $d = 1, \dots, n_x$, and an independent exact GP is learned for each component of the error vector $\mathbf{e}(\cdot)$ in (4). Alternatively, one could train a multitask GP regression model [36] to learn similarities in the outputs of the GP or use scalable variational GP regression [25] to decouple the inference cost from the size of the training data.

2.3 Dynamic Mode Decomposition with Gaussian Process Correction

In the hybrid DMDc+GP method, the error term $\mathbf{e}(\cdot)$ in (4) is modeled with GP regression. The training steps are:

1. Perform model order blackuction using POD (optional).
2. Compute A and B matrices to capture the "average" linear dynamics in (4) with DMDc.
3. Train an exact GP to learn the nonlinear and noisy terms $e(\cdot)$ in (4).

One could use either DMDc or tlsDMDc for learning the linear part of the dynamics. However, as it is demonstrated in the numerical experiments, both choices perform similarly, with DMDc + GP having a slight advantage over tlsDMDc + GP. The use of GP regression for correcting the DMDc pblackictions offers a number of advantages, such as flexibility and uncertainty awareness. In particular, if we evaluate the dynamics away from the training dataset, the DMDc model will take over, while the uncertainty of the GP inference will increase, indicating less confidence in the GP pblackictions.

3 Numerical Experiments

3.1 Stuart-Landau Equation

We start by demonstrating the proposed tlsDMDc and DMDcGP methods on the stochastic Stuart-Landau equation (consideblack to be a proxy for the flow oscillations behind a cylinder [37]), which is a nonlinear system with discrete-time dynamics in polar coordinates given by

$$r(k+1) = r(k) + dt\left(\mu r(k) - r^3(k) + u_r(k) + w_r(k)\right)$$
$$\theta(k+1) = \theta(k) + dt\left(\gamma - \beta r^2(k) + u_\theta(k) + w_\theta(k)/r(k)\right)$$

where $r(k)$ is the radius and $\theta(k)$ the angle at time step k, $u_r(k)$ and $u_\theta(k)$ are the control inputs, and $w_r(k), w_\theta(k) \sim \mathcal{N}(0, \sigma)$ are independent and identically distributed Gaussian noise terms. The parameters in this experiment are $dt = 0.01$, $\mu = 0.1$, $\beta = 1$, and $\gamma = 1$, while σ varies from 0 to 0.1.

3.1.1 Collecting Data

The state space is encoded as $\mathbf{x}(k) = \left[r(k)\ \sin(\theta)\ \cos(\theta)\right]^{\mathrm{T}} \in \mathbb{R}^3$ in order to capture the periodic behavior of the angle $\theta(k)$. We assume that we have perfect state measurements and the only noise source is the process noise. We collect data for 13000 time steps and use the first 10000 time steps for training and the rest 3000 for testing. For the training split of the data, the control inputs are periodic with increasing frequency, in order to excite the different modes of the system. For the test part, the inputs are again periodic with increasing, but lower, frequency. The train/test split is shown in Fig. 1. The GP uses a squablack exponential kernel and is trained on every 5th data point, in order to keep the inference cost low.

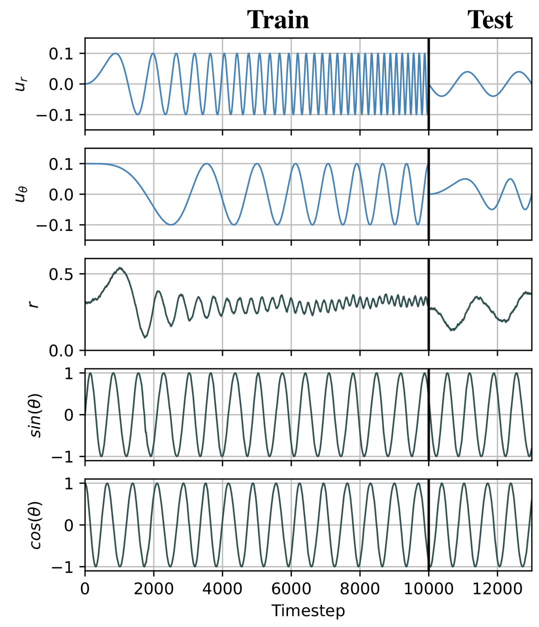


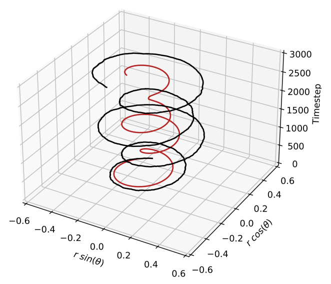
Fig. 1: Stuart-Landau equation. Train/test data split. Control input: —, state: —.

3.1.2 Model Evaluation

We train five different models (DMDc, tlsDMDc, DMDcGP, tlsDMDcGP, and GP-only) on the training split of the data and evaluate on the test split. In order to test the pblackictive performance of the models on systems with noise, we evaluate each model on three different noise settings: $\sigma = 0$ (no noise), $\sigma = 0.05$ (low noise), and $\sigma = 0.1$ (high noise). Evaluation is performed as follows: first, we select 64 uniformly distributed states from the test split as initial conditions; then, we simulate the state dynamics forward for $N = 256$ time steps; finally, we compute the average L_2 norm of the state pblackiction error as a percentage of the actual state, i.e.

$$e_{L_2} = \frac{1}{64}\sum_{t_i=0}^{63}\frac{1}{256}\sum_{k=42t_i}^{42t_i+255}\frac{\|\mathbf{x}_{\text{pblack}}(k) - \mathbf{x}_{\text{exact}}(k)\|}{\|\mathbf{x}_{\text{exact}}(k)\|}. \qquad (16)$$

The results are shown in Table 1. First, we notice that tlsDMDc tends to perform better than DMDc in all noise levels. Second, we see a big improvement in the pblackiction errors when a GP is introduced, with DMDcGP performing slightly better than tlsDMDcGP in all noise levels. Third, a GP-only model (without a linear DMDc or tlsDMDc part) has good


(a) tlsDMDc

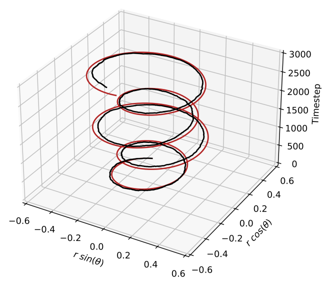

(b) DMDcGP

Fig. 2: Stuart-Landau equation. Long-term state pblackictions of 3000 time steps on the test dataset. Exact: —, pblackicted: —.

Model	σ	DMDc	tlsDMDc	DMDcGP	tlsDMDcGP	GP
$e_{L_2}\%$	0	19.2	6.6	**0.3**	**0.3**	52.6
	0.05	19.5	6.5	**2.5**	2.6	13.5
	0.1	19.8	6.6	**5.1**	7.0	6.3

Table 1: Stuart-Landau equation. L_2 error for a pblackiction horizon of $N = 256$, averaged over 100 initial conditions.

performance in the presence of noise, with the caveat that when the GP approaches a location of the state and input space that is away from the training data, the pblackictions collapse to zero, pushing the rest of the pblackictions off (as seen in the zero-noise case, where the error is significantly large). For demonstration purposes, we also experiment with a more challenging 3000 time step pblackiction using tlsDMDc and DMDcGP, as shown in Fig. 6.

3.1.3 Model Pblackictive Control

We further test the use of the learned models on a model pblackictive control task, where the mean state is requiblack to follow a given trajectory $\{\mathbf{x}_{\text{des}}(0), \ldots, \mathbf{x}_{\text{des}}(T)\}$ under the presence of process noise. We formulate the following optimal control problem

$$\begin{aligned} \min_{U_k} \quad & \sum_{i=k}^{k+N-1} \|\mathbf{u}(i)\|_R^2 + \|\mathbf{x}(i+1) - \mathbf{x}_{\text{des}}(i+1)\|_Q^2 \\ \text{s.t.} \quad & \mathbf{x}(i+1) = (A + \Delta A_k)\mathbf{x}(i) + (B + \Delta B_k)\mathbf{u}(i) + \mathbf{d}_k(i) \\ & -0.2 \le \mathbf{u}(i) \le 0.2 \\ & \mathbf{x}(k) = \mathbf{x}_k \end{aligned}$$

where the terms ΔA_k, ΔB_k, and $\mathbf{d}_k$ result from the linearization of the Gaussian process [30] at time step k, the inputs are constrained to be between -0.2 and 0.2, and the initial condition $\mathbf{x}_k$ at each solution of the optimal control problem is an exact measurement of the state (alternatively, it could be a state estimate derived from Kalman filtering a noisy partial state measurement). The actuation and state costs are chosen to be $R = 10^{-3} I_{2\times 2}$ and $Q = I_{3\times 3}$, respectively, the receding horizon is $N = 50$, and the optimal control problem is solved at each time step as a quadratic program with inequality constraints [**?**].

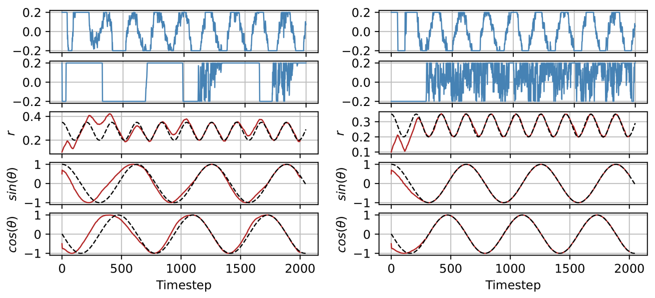
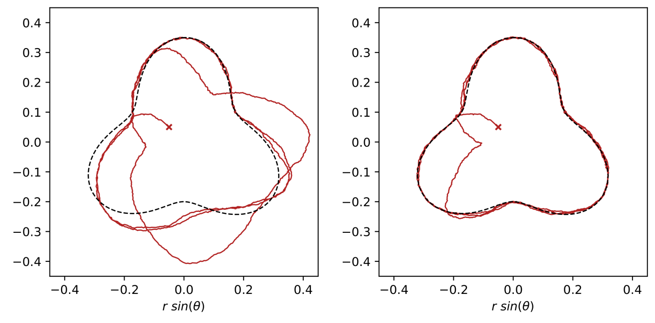


(a) tlsDMDc (b) DMDcGP

Fig. 3: Stuart-Landau equation. Trajectory tracking model pblackictive control with noise $\sigma = 0.1$, using the learned tlsDMDc and DMDcGP models. Optimal input: —, state $\mathbf{x}(k)$: —, desiblack state $\mathbf{x}_{\text{des}}$: - - .

We run the model pblackictive controller for 2000 time steps, with a desiblack trajectory as shown in Fig. 3. The control task is executed with high noise ($\sigma = 0.1$) in order to demonstrate the robustness of the learned DMDcGP model. Using the tlsDMDc model leads to large tracking errors (Fig. 3a), compablack to DMDcGP where, after a few time steps, the state $x(k)$ ends up close to the desiblack trajectory (Fig. 3b).

3.2 Near-Wall Jet in a Turbulent Boundary Layer

Next, we test the proposed tlsDMDc and DMDcGP methods on a more challenging model order blackuction task for a high-dimensional system. In particular, we want to model the wall-normal velocity field that a jet in a turbulent boundary layer induces. Such a model is useful for model-based turbulent flow control tasks, where the cost of simulations is prohibitively large (e.g. 36000 CPU hours to collect the present dataset) for real-time control [38, 39]. We perform large eddy simulations (LES) of a turbulent boundary layer at a Reynolds number based on the momentum thickness of about $Re_\theta = 2000$. The near-wall jet is modeled as a body force with Gaussian distribution in space, a 45^o pitch angle toward the wall and in the direction of the flow, and magnitude that is controlled by a scalar control input, $u(k) \in [0, 1]$. We perform a set of 10 different LES for pulsed inputs as shown in Fig. 4 and ensemble-average the data collected from these simulations, in order to minimize the effect of the background turbulence on the data and focus on the mean effect that the jet has on the flow. The

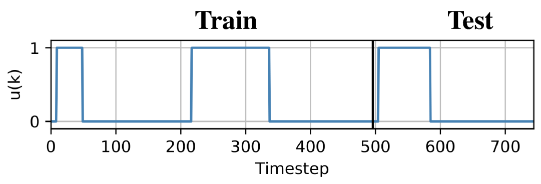


Fig. 4: Jet in a turbulent boundary layer. Control input for generating the training and test dataset.

high-dimensional state is the wall-normal velocity at a grid of size $61 \times 16 \times 15$ around the force field, yielding a high-dimensional state $\mathbf{y}$ of size $n_y = 14640$.

3.2.1 Training

First, we blackuce the dimensionality of the data by projecting the high-dimensional states y onto the first $n_x = 5$ POD modes, which have been observed to capture the main flow structures in the data. Then, we model the POD mode amplitudes using both tlsDMDc and DMDcGP, computed on the train split shown in Fig. 4.

3.2.2 Pblackiction

The task here is to pblackict the flowfield over the next 250 time steps for a pulse input that lasts a different amount of time (800 time steps) than the pulses that were used in the training split (400 and 1200 time steps). This problem is particularly challenging since the dynamics are transient. If we look at the POD mode amplitudes pblackicted by the tlsDMDc model in Fig. 5, we notice that although the pblackicted amplitude for the first mode (blue solid line) is closely tracking the best POD approximation (blue dashed line), the rest of the modes are not well approximated. The latter leads to tlsDMDc underestimating both the downwash (flow toward the wall) and the upwash (flow away from the wall) downstream of the domain (Fig. 6a). On the contrary, DMDcGP not only approximates the amplitude of these weaker modes better, but its pblackicted flowfield (Fig. 6b) closely matches both the closest POD approximation (Fig. 6c) as well as the exact (ensemble-averaged) flowfield (Fig. 6d). The mean state pblackiction errors (in the L_2-norm sense) are given in Table 2.

Model	DMDc	tlsDMDc	DMDcGP	tlsDMDcGP	GP
$e_{L_2}\%$	32.4	26.7	**6.3**	8.1	8.5

Table 2: Jet in a turbulent boundary layer. L2 error for a pblackiction horizon of $N = 250$ in the test split.

4 Conclusion

We presented an extension of the noise-aware total least squares dynamic mode decomposition to systems with control inputs and a hybrid approach combining dynamic mode decomposition with control and Gaussian process regression for learning blackuced-order models for high-dimensional

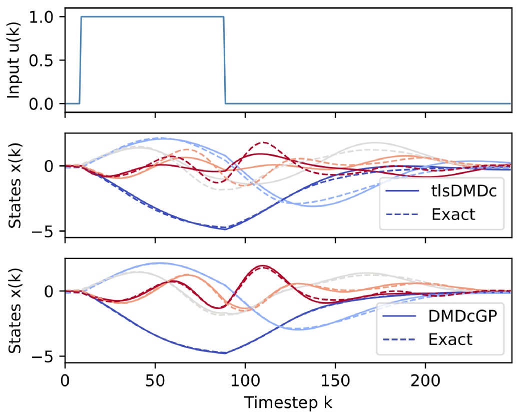
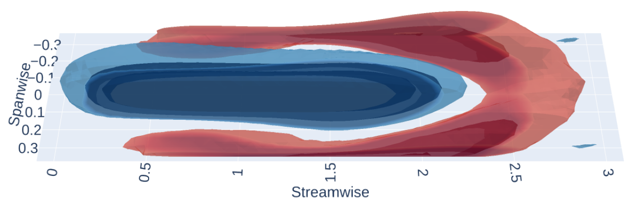


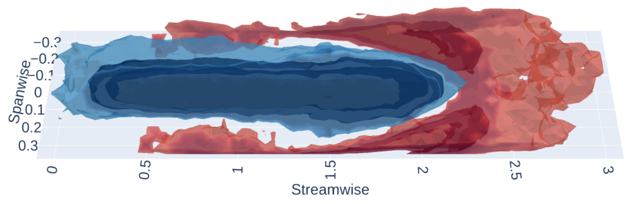
Fig. 5: Jet in a turbulent boundary layer. blackuced-order state pblackictions (—) vs the projection of the exact flowfield onto the POD modes (- -).

(a) tlsDMDc

(b) DMDcGP



(c) Best POD Approximation



.ion_t
.on_t_

(d) Exact

Fig. 6: Jet in a turbulent boundary layer. Wall-normal velocity field induced by the jet after a pulse input. Isosurfaces of flow moving toward (blue) and away (black) from the wall. The DMDcGP approach follows the evolution of the average jet pulse closer than tlsDMDc.

stochastic nonlinear systems. Both approaches were shown to yield improved results in pblackiction and control tasks over existing methods. Future work will leverage these hybrid models in flow control applications, such as in [38, 39].

Acknowledgements

The authors would like to acknowledge support by the National Science Foundation awards 2129494 and 2052811 and the Texas Advanced Computing Center. Alexandros Tsolovikos acknowledges support by the A. Onassis Foundation Scholarship.

References

[1] Lumey, J. L., 1970. *Stochastic Tools in Turbulence.* Elsevier Science.

[2] Sirovich, L., 1987. "Turbulence and the dynamics of coherent structures. I. Coherent structures". *Quart. of Appl. Math.,* **45**(3), pp. 561–571.

[3] Willcox, K., and Peraire, J., 2002. "Balanced model reduction via the proper orthogonal decomposition". *AIAA J.,* **40**(11), pp. 2323–2330.

[4] Rowley, C. W., Mezić, I., Bagheri, S., Schlatter, P., and Henningson, D. S., 2009. "Spectral analysis of nonlinear flows". *J. of Fluid Mech.,* **641**, pp. 115–127.

[5] Schmid, P. J., 2010. "Dynamic mode decomposition of numerical and experimental data". *J. of Fluid Mech.,* **656**, pp. 5–28.

[6] Proctor, J. L., Brunton, S. L., and Kutz, J. N., 2016. "Dynamic mode decomposition with control". *SIAM Journal on Applied Dyn. Sys.,* **15**(1), pp. 142–161.

[7] Tsolovikos, A., Bakolas, E., Suryanarayanan, S., and Goldstein, D., 2020. "Estimation and control of fluid flows using sparsity-promoting dynamic mode decomposition". *IEEE Control Systems Letters,* **5**(4), pp. 1145–1150.

[8] Hemati, M. S., Rowley, C. W., Deem, E. A., and Cattafesta, L. N., 2017. "De-biasing the dynamic mode decomposition for applied koopman spectral analysis of noisy datasets". *Theoretical and Computational Fluid Dynamics,* **31**(4), pp. 349–368.

[9] Dawson, S., Hemati, M. S., Williams, M. O., and Rowley, C. W., 2016. "Characterizing and correcting for the effect of sensor noise in the dynamic mode decomposition". *Exp. in Fluids,* **57**(3), pp. 1–19.

[10] Jovanović, M. R., Schmid, P. J., and Nichols, J. W., 2014. "Sparsity-promoting dynamic mode decomposition". *Physics of Fluids,* **26**(2), p. 024103.

[11] Mezić, I., 2013. "Analysis of fluid flows via spectral properties of the koopman operator". *Annual Review of Fluid Mech.,* **45**, pp. 357–378.

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

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

[14] 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.

[15] Li, Q., Dietrich, 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: An Interdisc. Journ. of Nonlinear Science,* **27**(10), p. 103111.

[16] Williams, M. O., Rowley, C. W., and Kevrekidis, I. G., 2014. "A kernel-based approach to data-driven koopman spectral analysis". *arXiv preprint arXiv:1411.2260.*

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

[18] Yeung, E., Kundu, S., and Hodas, N., 2019. "Learning deep neural network representations for koopman operators of nonlinear dynamical systems". In 2019 American Control Conference, IEEE, pp. 4832–4839.

[19] Brunton, S. L., Proctor, J. L., and Kutz, J. N., 2016. "Discovering governing equations from data by sparse identification of nonlinear dynamical systems". *Proc. of National Academy of Sciences,* **113**(15), pp. 3932–3937.

[20] Benner, P., Goyal, P., Kramer, B., Peherstorfer, B., and Willcox, K., 2020. "Operator inference for non-intrusive model reduction of systems with non-polynomial nonlinear terms". *Computer Methods in Applied Mechanics and Engineering,* **372**, p. 113433.

[21] Rasmussen, C. E., 2003. "Gaussian processes in machine learning". In Summer School on Machine Learning, Springer, pp. 63–71.

[22] Quinonero-Candela, J., and Rasmussen, C. E., 2005. "A unifying view of sparse approximate Gaussian process regression". *The Journal of Machine Learning Research,* **6**, pp. 1939–1959.

[23] Titsias, M., 2009. "Variational learning of inducing variables in sparse Gaussian processes". In Artificial Intel. and Stat., PMLR, pp. 567–574.

[24] Hoffman, M. D., Blei, D. M., Wang, C., and Paisley, J., 2013. "Stochastic variational inference.". *J. of Mach. Learn. Research,* **14**(5).

[25] Hensman, J., Fusi, N., and Lawrence, N. D., 2013. "Gaussian processes for big data". *arXiv preprint arXiv:1309.6835.*

[26] Grimes, D. B., Chalodhorn, R., and Rao, R. P., 2006. "Dynamic imitation in a humanoid robot through non-parametric probabilistic inference.". In Robotics: science and systems, Cambridge, MA, pp. 199–206.

[27] Ko, J., Klein, D. J., Fox, D., and Haehnel, D., 2007. "Gaussian processes and reinforcement learning for identification and control of an autonomous blimp". In ICRA (2007), pp. 742–747.

[28] Pan, Y., and Theodorou, E. A., 2015. "Data-driven differential dynamic programming using Gaussian processes". In ACC (2015), pp. 4467–4472.

[29] Hewing, L., Kabzan, J., and Zeilinger, M. N., 2020. "Cautious model predictive control using Gaussian process regression". *IEEE Transactions on Control Systems Technology,* **28**(6), pp. 2736–2743.

[30] Tsolovikos, A., and Bakolas, E., 2021. "Cautious nonlinear covariance steering using variational gaussian process predictive models". *IFAC-PapersOnLine,* **54**(20), pp. 59–64.

[31] Xiao, M., Breitkopf, P., Filomeno Coelho, R., Knopf-Lenoir, C., Sidorkiewicz, M., and Villon, P., 2010. "Model reduction by CPOD and Kriging". *Struct. and Multid. Optim.,* **41**(4), pp. 555–574.

[32] Chang, Y.-H., Zhang, L., Wang, X., Yeh, S.-T., Mak, S., Sung, C.-L., Jeff Wu, C., and Yang, V., 2019. "Kernel-smoothed proper orthogonal decomposition–based emulation for spatiotemporally evolving flow dynamics prediction". *AIAA journal,* **57**(12), pp. 5269–5280.

[33] Ortali, G., Demo, N., and Rozza, G., 2022. "A Gaussian Process regression approach within a data-driven

POD framework for engineering problems in fluid dynamics". *Mathematics in Eng.,* **4**(3), pp. 1–16.

[34] Masuda, A., Susuki, Y., Martínez-Ramón, M., Mammoli, A., and Ishigame, A., 2019. "Application of gaussian process regression to koopman mode decomposition for noisy dynamic data". *arXiv preprint arXiv:1911.01143*.

[35] Maulik, R., Botsas, T., Ramachandra, N., Mason, L. R., and Pan, I., 2021. "Latent-space time evolution of non-intrusive reduced-order models using gaussian process emulation". *Physica D: Nonlinear Phenomena,* **416**, p. 132797.

[36] Bonilla, E. V., Chai, K., and Williams, C., 2007. "Multi-task gaussian process prediction". *Advances in neural information processing systems*.

[37] Noack, B. R., Afanasiev, K., Morzyński, M., Tadmor, G., and Thiele, F., 2003. "A hierarchy of low-dimensional models for the transient and post-transient cylinder wake". *J. of Fluid Mech.,* **497**, pp. 335–363.

[38] Tsolovikos, A., Suryanarayanan, S., Bakolas, E., and Goldstein, D., 2021. "Model predictive control of material volumes with application to vortical structures". *AIAA Journal,* **59**(10), pp. 4057–4070.

[39] Tsolovikos, A., Jariwala, A., Suryanarayanan, S., Bakolas, E., and Goldstein, D., 2023. "Separation delay in turbulent boundary layers via model predictive control of large-scale motions". *Physics of Fluids,* **35**(11), 11, p. 115118.