Data-Efficient Model Learning for Model Predictive Control with Jacobian-Regularized Dynamic Mode Decomposition

Anonymous Author(s) Affiliation Address email

Abstract: We present a data-efficient algorithm for learning models for model-1 predictive control (MPC). Our approach, Jacobian-Regularized DMD (JDMD), 2 offers improved sample efficiency over traditional Koopman approaches based on 3 Dynamic-Mode Decomposition (DMD) by leveraging Jacobian information from 4 5 an approximate prior model of the system, and improved tracking performance 6 over traditional model-based MPC. We demonstrate JDMD's ability to quickly learn bilinear Koopman dynamics representations across several realistic exam-7 ples in simulation, including a perching maneuver for a fixed-wing aircraft with 8 an experimentally derived high-fidelity physics model. In all cases, we show that 9 the models learned by JDMD provide superior tracking and generalization perfor-10 11 mance in the presence of significant model mismatch within a model-predictive control framework, when compared to the approximate prior models used in train-12 ing and models learned by standard extended DMD. 13

14 **1** Introduction

In recent years, both model-based optimal-control [1, 2, 3, 4] and data-driven reinforcement-learning 15 methods [5, 6, 7] have demonstrated impressive successes on complex, nonlinear robotic systems. 16 However, both approaches suffer from inherent drawbacks: Data-driven methods often require ex-17 tremely large amounts of data and fail to generalize outside of the domain or task on which they were 18 trained. On the other hand, model-based methods require an accurate model of the system to achieve 19 good performance. In many cases, high-fidelity models can be too difficult to construct from first 20 principles or too computationally expensive to be of practical use. However, low-order approximate 21 models that can be evaluated cheaply at the expense of controller performance are often available. 22 With this in mind, we seek a middle ground between model-based and data-driven approaches in 23 this work. 24

We propose a method for learning bilinear Koopman models of nonlinear dynamical systems for use 25 in model-predictive control that leverages derivative information from an approximate prior dynam-26 27 ics model of the system in the training process. Given the increased availability of differentiable simulators [8, 9], this approximate derivative information is readily available for many systems of 28 interest. Our new algorithm builds on extended Dynamic Mode Decomposition (EDMD), which 29 learns Koopman models from trajectory data [10, 11, 12, 13, 14], by adding a derivative regular-30 ization term based on derivatives computed from a prior model. We show that this new algorithm, 31 Jacobian-regularized Dynamic Mode Decomposition (JDMD), can learn models with dramatically 32 fewer samples than EDMD, even when the prior model differs significantly from the true dynamics 33 of the system. We also demonstrate the effectiveness of these learned models in a model-predictive 34 control (MPC) framework. The result is a fast, robust, and sample-efficient pipeline for quickly train-35 ing a model that can outperform MPC controllers using the approximate analytical model as well 36 models learned using both traditional Koopman approaches and multi-layer perceptrons (MLPs). 37

Submitted to the 6th Conference on Robot Learning (CoRL 2022). Do not distribute.

38 While our proposed Koopman-based approached is significantly more sample efficient, we also

demonstrate the utility of incorporating gradient information for learning a simple model using a two lower MLP.

40 two-layer MLP.

Our work is most closely related to the recent work of Folkestad et. al. [13, 15, 16], which learn
bilinear models and apply nonlinear model-predictive control directly on the learned bilinear dynamics. Other recent works have combined linear Koopman models with model-predictive control
[12] and Lyapunov control techniques with bilinear Koopman [17]. Our contributions are:

A novel extension to extended dynamic mode decomposition, called JDMD, that incorporates gradient information from an approximate analytic model

A recursive, batch QR algorithm for solving the least-squares problems that arise when
 learning bilinear dynamical systems using DMD-based algorithms, including JDMD and
 EDMD

The remainder of the paper is organized as follows: In Section 2 we provide some background 50 on the application of Koopman operator theory to controlled dynamical systems and review some 51 related works. Section 3 then describes the proposed JDMD algorithm. In Section 4 we outline 52 a memory-efficient technique for solving the large, sparse linear least-squares problems that arise 53 when applying JDMD and other DMD-based algorithms. Section 5 then provides simulation results 54 and analysis of the proposed algorithm applied to control tasks on a cartpole, a quadrotor, and a small 55 foam airplane with an experimentally determined aerodynamics model, all subject to significant 56 57 model mismatch. It also includes a comparison of the current approach to model-learning via a multi-layer perceptron, for the canonical cartpole problem. In Section 6 we discuss the limitations 58 of our approach, followed by some concluding remarks in Section 7. 59

60 2 Background and Related Work

61 2.1 Koopman Operator Theory

⁶² The theoretical underpinnings of the Koopman operator and its application to dynamical systems has

⁶³ been extensively studied [18, 19, 11, 20, 21]. Rather than describe the theory in detail, we highlight

the key concepts employed by the current work and refer the reader to the existing literature onKoopman theory for further details.

⁶⁶ We start by assuming a controlled, nonlinear, discrete-time dynamical system,

$$x^+ = f(x, u),\tag{1}$$

where $x \in \mathcal{X} \subseteq \mathbb{R}^{N_x}$ is the state vector, $u_k \in \mathbb{R}^{N_u}$ is the control vector, and x^+ is the state at the next time step. Assuming the dynamics are control-affine, the nonlinear finite-dimensional system (1) can be represented *exactly* by an infinite-dimensional bilinear system through the Koopman canonical transform [21]. This bilinear Koopman model follows the form,

$$y^{+} = Ay + Bu + \sum_{i=1}^{m} u_{i}C_{i}y = g(y, u),$$
(2)

where $y = \phi(x)$ is a nonlinear mapping from the finite-dimensional state space \mathcal{X} to the infinitedimensional Hilbert space of *observables* \mathcal{Y} . In practice, we approximate (2) by restricting \mathcal{Y} to be a finite-dimensional vector space, in which case ϕ becomes a finite-dimensional nonlinear function of the state variables, which can be either chosen heurstically based on domain expertise, or learned [22, 23, 24].

Intuitively, ϕ "lifts" our state x into a higher dimensional space \mathcal{Y} where the dynamics are approximately (bi)linear, effectively trading dimensionality for (bi)linearity. Similarly, we can perform an

rs since we embed the original state within the nonlinear mapping [11, 15, 25, 26, 27], ϕ is constructed

⁸⁰ in such a way that this unlifting is linear:

$$x = Gy. \tag{3}$$

We note that our proposed method does not rely on this assumption: any mapping could be used. 81 The problem of finding an optimal mapping is itself a major area of research, and many recent 82 studies have focused on jointly learning both the model and the mapping [22, 23, 28, 29, 24]. While 83 clearly advantageous, learning an optimal embedding is orthogonal to the main focus of the current 84 paper, which focuses on a straightforward way of incorporating analytical derivative information 85 from an approximate model, which is equally applicable whether the embedding function is learned 86 or chosen heuristically. The mappings in the current work are chosen heuristically based on problem 87 insight and experience. 88

89 2.2 Extended Dynamic Mode Decomposition

A lifted bilinear system of the form (2) can be learned from P samples of the system dynamics (x_j^+, x_j, u_j) using Extended Dynamic Mode Decomposition (EDMD) [20, 15]. We first define the following data matrices:

$$Z_{1:P} = \begin{bmatrix} y_1 & y_2 & \dots & y_P \\ u_1 & u_2 & \dots & u_P \\ u_{1,1}y_1 & u_{2,1}y_2 & \dots & u_{P,1}y_P \\ \vdots & \vdots & \ddots & \vdots \\ u_{1,m}y_1 & u_{2,m}y_2 & \dots & u_{P,m}y_P \end{bmatrix}, \quad Y_{1:P}^+ = \begin{bmatrix} y_1^+ & y_2^+ & \dots & y_P^+ \end{bmatrix}, \quad (4)$$

⁹³ We then concatenate all of the model coefficient matrices as follows:

$$E = \begin{bmatrix} A & B & C_1 & \dots & C_m \end{bmatrix} \in \mathbb{R}^{N_y \times N_z},\tag{5}$$

⁹⁴ The model learning problem can then be written as the following linear least-squares problem:

$$\underset{E}{\text{minimize}} \|EZ_{1:P} - Y_{1:P}^{+}\|_{2}^{2}$$
(6)

95 EDMD is closely related to classical feature-based machine learning approaches like the "kernel

trick" used in support vector machines [30], but extends these ideas to bilinear models of controlled

97 dynamical systems.

38 3 Jacobian-Regularizated Dynamic Mode Decomposition

We now present JDMD as a straightforward adaptation of the original EDMD algorithm described in Section 2.2. Given P samples of the dynamics (x_i^+, x_i, u_i) , and an approximate discrete-time dynamics model,

$$x^+ = \tilde{f}(x, u),\tag{7}$$

we can evaluate the Jacobians of our approximate model \tilde{f} at each of the sample points: $\tilde{A}_i = \frac{\partial \tilde{f}}{\partial x}$, $\tilde{B}_i = \frac{\partial \tilde{f}}{\partial u}$. After choosing a nonlinear mapping $\phi : \mathbb{R}^{N_x} \mapsto \mathbb{R}^{N_y}$ our goal is to find a bilinear dynamics model (2) that matches the Jacobians of our approximate model, while also matching our dynamics samples. We accomplish this by penalizing differences between the Jacobians of our learned bilinear model with respect to the original states x and controls u, and the Jacobians we expect from our analytical model. These *projected Jacobians* are calculated by differentiating through the *projected dynamics*:

$$x^{+} = G\left(A\phi(x) + Bu + \sum_{i=1}^{m} u_i C_i \phi(x)\right) = \bar{f}(x, u).$$
(8)

109 Differentiating (8) with respect to x and u gives us

$$\bar{A}_j = \frac{\partial \hat{f}}{\partial x}(x_j, u_j) = G\left(A + \sum_{i=1}^m u_{j,i}C_i\right)\Phi(x_j) = GE\hat{A}(x_j, u_j) = GE\hat{A}_j$$
(9a)

$$\bar{B}_j = \frac{\partial f}{\partial u}(x_j, u_j) = G\Big(B + \begin{bmatrix} C_1 x_j & \dots & C_m x_j \end{bmatrix}\Big) = GE\hat{B}(x_j, u_j) = GE\hat{B}_j$$
(9b)

110 where $\Phi(x) = \partial \phi / \partial x$ is the Jacobian of the nonlinear map ϕ , and

$$\hat{A}(x,u) = \begin{bmatrix} I_{N_y} \\ 0 \\ u_1 I_{N_y} \\ u_2 I_{N_y} \\ \vdots \\ u_m I_{N_y} \end{bmatrix} \Phi(x) \in \mathbb{R}^{N_z \times N_x}, \quad \hat{B}(x,u) = \begin{bmatrix} 0 \\ I_{N_u} \\ [\phi(x) \ 0 \ \dots \ 0] \\ [0 \ \phi(x) \ \dots \ 0] \\ \vdots \\ [0 \ 0 \ \dots \ \phi(x)] \end{bmatrix} \in \mathbb{R}^{N_z \times N_u}.$$
(10)

111 We then solve the following linear least-squares problem:

minimize
$$(1-\alpha) \|EZ_{1:P} - Y_{1:P}^+\|_2^2 + \alpha \sum_{j=1}^P \left(\|GE\hat{A}_j - \tilde{A}_j\|_2^2 + \|GE\hat{B}_j - \tilde{B}_j\|_2^2 \right)$$
 (11)

The resulting linear least-squares problem has $(N_y + N_x^2 + N_x \cdot N_u) \cdot P$ rows and $N_y \cdot N_z$ columns. Given that the number of rows in this problem grows quadratically with the state dimension, solving this problem can be challenging from a computational perspective. In the Section 4, we propose an algorithm for solving these problems without needing to move to a distributed-memory setup in order to solve these large linear systems. The proposed method also provides a straightforward way to approach incremental updates to the bilinear system, where the coefficients could be efficiently learned "live" while the robot gathers data by moving through its environment.

119 4 Efficient Recursive Least Squares

¹²⁰ In its canonical formulation, a linear least squares problem can be represented as the following ¹²¹ unconstrained optimization problem:

$$\min_{x} \|Fx - d\|_2^2. \tag{12}$$

We assume F is a large, sparse matrix and that solving it directly using a QR or Cholesky decomposition requires too much memory for a single computer. While solving (12) using an iterative method such as LSMR [31] or LSQR [32] is possible, we find that these methods do not work well in practice for solving (11) due to ill-conditioning. Standard recursive methods for solving these problems are able to process the rows of the matrices sequentially to build a QR decomposition of the full matrix, but also tend to suffer from ill-conditioning [33, 34, 35].

To overcome these issues, we propose an alternative recursive method based. We solve (12) by dividing up rows of F into batches:

$$F^{T}F = F_{1}^{T}F_{1} + F_{2}^{T}F_{2} + \ldots + F_{N}^{T}F_{N}.$$
(13)

The main idea is to maintain and update an upper-triangular Cholesky factor U_i of the first *i* terms of the sum (13). Given U_i , we can calculate U_{i+1} using the QR decomposition, as shown in [36]:

$$U_{i+1} = \sqrt{U_i^T U_i + F_{i+1}^T F_{i+1}} = \operatorname{QR}_{\mathrm{R}} \left(\begin{bmatrix} U_i \\ F_{i+1} \end{bmatrix} \right), \tag{14}$$

where QR_R returns the upper triangular matrix R from the QR decomposition. For an efficient implementation, this function should be an "economy" or "Q-less" QR decomposition since the Qmatrix is never needed. 135 We also handle regularization of the normal equations, equivalent to adding Tikhonov regularization

to the original least squares problem, during the base case of our recursion. If we want to add an L2

regularization with weight λ , we calculate U_1 as:

$$U_1 = \text{QR}_{\text{R}} \left(\begin{bmatrix} F_1\\ \sqrt{\lambda}I \end{bmatrix} \right).$$
(15)

Throughout the paper, the results presented for both EDMD and JDMD correspond to the bestperforming L2-regularization values for each algorithm to ensure a fair comparison is made. We perform a sweep over a range of L2-regularization values for each study, with MPC tracking error as the metric.

142 **5 Experimental Results**

This section presents the results of several simulation experiments to evaluate the performance of JDMD. For each simulated system we specify two models: a *nominal* model, which is simplified and contains both parametric and non-parametric model error, and a *true* model, which is used exclusively for simulating the system and evaluating algorithm performance.

All models were trained by simulating the "true" system with a nominal controller to collect data in 147 the region of the state space relevant to the task. A set of fixed-length trajectories were collected, 148 each at a sample rate of 20-25 Hz. The bilinear EDMD model was trained using the same approach 149 introduced by Folkestad and Burdick [15]. When applying MPC to the learned Koopman models, the 150 projected Jacobians (9) were used, since this projected system is much more likely to be controllable 151 than the lifted one and reduces the computational complexity back to that of the nominal MPC 152 controller. This results in a nonlinear model in the original state space, which is linearized about 153 the reference trajectory to create a linear MPC controller. All continuous dynamics were discretized 154 with an explicit fourth-order Runge Kutta integrator. Code for all experiments is available at TODO: 155 removed for anonymous review. 156

157 5.1 Systems and Tasks

Cartpole: We perform a swing-up task on a cartpole system. The *true* model includes Coulomb 158 friction between the cart and the floor, viscous damping at both joints, and a deadband in the 159 control input that were not included in the nominal model. Additionally, the mass of the cart 160 and pole model were altered by 20% and 25% with respect to the nominal model, respec-161 tively. The following nonlinear mapping was used when learning the bilinear models: $\phi(x) =$ 162 $[1, x, \sin(x), \cos(x), \sin(2x), \sin(4x), T_2(x), T_3(x), T_4(x)] \in \mathbb{R}^{33}$, where $T_i(x)$ is a Cheby-163 shev polynomial of the first kind of order *i*. All reference trajectories for the swing up task were 164 generated using ALTRO [36, 37]. 165

Quadrotor: We track point-to-point linear reference trajectories from various initial condi-166 tions on both planar and full 3D quadrotor models. For both systems, the true model in-167 cludes aerodynamic drag terms not included in the nominal model, as well as parametric er-168 ror of roughly 5% on the system parameters (e.g. mass, rotor arm length, etc.). The planar 169 model was trained using a nonlinear mapping of $\phi(x) = [1, x, \sin(x), \cos(x), \sin(2x), T_2(x)] \in$ 170 \mathbb{R}^{25} while the full quadrotor model was trained using a nonlinear mapping of $\phi(x)$ = 171 $[1, x, T_2(x), \sin(p), \cos(p), R^T v, v^T R R^T v, p \times v, p \times \omega, \omega \times \omega] \in \mathbb{R}^{44}$, where p is the quadro-172 tor's position, v and ω are the translational and angular velocities respectively, and R is the rotation 173 matrix. 174

Airplane: We perform a post-stall perching maneuver on a high-fidelity model of a fixed-wing airplane. The perching trajectory is produced using trajectory optimization (see Figure 1a) and tracked using MPC. Perching involves flight at high angles of attack, where the aerodynamic lift and drag forces are extremely complex and difficult to model from first principles. We look to previous works where the simulated aerodynamics were fitted using empirical data from in-person,



(a) Expert perching demonstration, a high angle-of-attack maneuver that minimizes velocity at the goal position with complex, post-stall aerodynamic forces



(b) E-Flite AS3Xtra airplane model used in hardware data collection



(c) Experiment setup configurations for collecting flight data

Figure 1: Complex dynamics of a perching fixed-wing airplane. High-angle-of-attack perching maneuvers (top) require the modeling of complex post-stall aerodynamic effects. The simulated aerodynamic forces were modeled as functions using flight data collected from real-world hardware experiments (bottom).

wind-tunnel experiments (see Figure 1b and 1c) before being demonstrated on hardware platforms [38, 39]. The *true* model includes the empirically-modeled, nonlinear flight dynamics [39], while the *nominal* model uses a simple flat-plate wing model with linear lift and quadratic drag coefficient approximations. The bilinear models use a 68-dimensional nonlinear mapping ϕ including terms such as the rotation matrix (expressed in terms of a Modified Rodriguez Parameter), powers of the angle of attack and side slip angle, the body frame velocity, various cross products with the angular velocity, and some 3rd and 4th order Chebyshev polynomials of the states.

187 5.2 Sample Efficiency

We compare the sample efficiency of several algorithms on the cartpole swing-up task in Fig. 2, 188 including a simple two-layer multi-layer perceptron trained using the a loss function equivalent to 189 (11) with $\alpha = 1$ (MLP) and $\alpha \in (0,1)$ (JMLP). For JMLP, α was monotonically decreased over 190 time, in order to place more weight on the data as more data was used (red line in Fig. 2b). The 191 derivatives of the model with respect to the inputs are calculated automatically using backward prop-192 agation of the partial derivatives for usage in the loss function, resulting in second-order derivatives 193 of the tanh activation functions when calculating the gradient with respect to the model parameters. 194 As shown, the proposed method achieves the best performance overall, and does so with only two 195 training trajectories. In comparison, traditional EDMD requires about 10 iterations to achieve con-196 sistent performance, whereas the MLP methods require hundreds of training trajectories. It's also 197 important to note that by applying the proposed approach to an MLP we were able to dramatically 198 improve both the performance and sample efficiency of the MLP-based approach. Similar results 199 were obtained for the airplane perching example (Fig. 6c), where EDMD requires about 3x the 200 number of samples (35 vs 10) compared to the proposed approach, and never achieves the same 201 closed-loop performance. 202

203 5.3 Generalization

We demonstrate the generalizability of the proposed method on both the planar and 3D quadrotor. In all tasks, the goal is to return to the origin, given an initial condition sampled from some uniform distribution centered at the origin. To test the generalizability of the algorithms, we scale the size



Figure 2: Cartpole swingup MPC tracking error vs training trajectories for Koopman methods (left) and a multi-layer perceptron (right). The sample efficiency of both methods is significantly improved when derivative information is included in the loss function. Note that Koopman approaches require an order of magnitude fewer trajectories to stabilize compared the MLP-based approach. The median



error is shown as a thick line, while the shaded regions represent the 5% to 95% percentile bounds

(a) LQR stabilization error over increasing equilibrium offset for 100 random initial conditions.

on the 10 test trajectories.

(b) MPC Tracking error over increasing scope of test distribution for 50 random initial conditions.

Figure 3: Generalizability with respect to final or initial conditions sampled outside of the training domain, studied on planar quadrotor performing an LQR stabilization (left) and MPC tracking task (right). For the stabilization task, 100 equilibrium positions are sampled uniformly within an offset value. For the tracking task, 50 initial conditions are sampled from a uniform distribution, whose limits are determined by a scaling of those of the training distribution. A training range fraction greater than 1 (vertical gray dashed line) indicates the distribution range is beyond that used to generate the training trajectories. The median error is shown as a thick line, while the shaded regions represent the 5% to 95% percentile bounds.

of the sampling "window" relative to the window on which it was trained, e.g. if the initial lateral 207 position was trained on data in the interval [-1.5, +1.5], we sampled the test initial condition from 208 the window $[-\gamma 1.5, +\gamma 1.5]$. The results for the planar quadrotor are shown in Figure 3b, with γ up 209 to 2.5. As shown, JDMD generalizes well outside of the training window, where the performance 210 of EDMD varies significantly even within the training window, as shown by the growing region that 211 bounds the 5% to 95% percentile of the tracking performance over the 50 test cases. Additionally, 212 in Figure 3a we show the effect of changing the equilibrium position away from the origin: while 213 the true dynamics should be invariant to this change, EDMD fails to learn this whereas JDMD does. 214

For the full quadrotor, given the goal of tracking a straight line back to the origin, we test 50 initial conditions, many of which are far from the goal, have large velocities, or are nearly inverted (see Figure 5a). The results using an MPC controller are shown in Table 1, demonstrating the excellent generalizability of the algorithm, given that the algorithm was only trained on 30 initial conditions, sampled relatively sparsely given the size of the sampling window. EDMD only successfully brings about 18% of the samples to the origin, while the majority of the time resulting in trajectories like



Figure 4: Loss versus number of training trajectories for the cartpole MLP. Although the both models perform about equally well on instantaneously predicting the discrete dynamics, the sample efficiency and performance on the closed-loop control problem different significantly (see Figure 2b.)



(a) Generated point-to-point trajectories and initial conditions for testing tracking MPC of 6-DOF quadro-tor.

(b) Performed trajectories of nominal MPC (black), EDMD (orange), and JDMD (cyan) for tracking infeasible, point-to-point trajectory (red).

Figure 5: Point-to-point, test trajectory generation and example tracking performance of full, 6-DOF quadrotor. The test trajectories generated include a wide scope of initial conditions beyond that of the training set, such as high position offset, large velocities, and near-inverted attitude. JDMD often had the best tracking performance while successfully reaching the goal state, with a similar success rate as nominal MPC within a tighter distribution.

those in Figure 5b. JDMD improves the tracking performance of nominal MPC, which is subject to a constant error bias due to model mismatch, as shown in Fig. 5b.

223 5.4 Sensitivity to Model Mismatch

While we've introduced a significant mount of 224 model mismatch in all of the examples so far, 225 a natural argument against model-based meth-226 ods is that they're only as good as your model 227 is at capturing the salient dynamics of the sys-228 tem. We investigated the effect of increasing 229 model mismatch by incrementally increasing 230 the Coulomb friction coefficient between the 231 cart and the floor for the cartpole stabilization 232 task (recall the nominal model assumed zero 233

	Nominal	EDMD	JDMD
Success Rate	82%	18%	80%
Median	0.30	0.63	0.11
5% Quantile	0.13	0.08	0.03
95% Quantile	0.38	2.62	0.23

Table 1: Performance summary of MPC tracking of 6-DOF quadrotor. Other than success rate, all values are the tracking error of the successfully stabilized trajectories.

friction). The results are shown in Table 2. As expected, the number of training trajectories required to find a good stabilizing controller increases for the proposed approach. We achieved the results above by setting $\alpha = 0.01$, corresponding to a decreased confidence in our model, thereby

Friction (μ)	0.0	0.1	0.2	0.3	0.4	0.5	0.6
Nominal	1	1	X	X	X	X	X
EDMD	3	19	6	14	X	X	X
JDMD	2	2	2	2	3	7	12

Table 2: Training trajectories required to stabilize the cartpole with the given friction coefficient



(a) Median JDMD model prediction error (openloop) and MPC tracking error (closed-loop) for perching airplane over varying α values. Closedloop behavior changes little with respect to open-loop prediction error. The missing open-loop values are points there the states of the open-loop system diverged to infinity.

(b) Error from true model Jacobians for the nominal model, EDMD, and JDMD. With just a few training trajectories, JDMD closely matches the Jacobian information from the nominal model. Even with substantial training data EDMD has significant error in the Jacobians.

(c) Sample efficiency for the airplane perching problem. JDMD learns the model with only 10 training trajectories, whereas EDMD requires about 35. Both models perform significantly better than nominal MPC due to significant model mismatch at high angles of attack.

Figure 6: Results on the airplane perching task

placing greater weight on the experimental data. The standard EDMD approach always required more samples, and was unable to find a good enough model above friction values of 0.4. While this could likely be remedied by adjusting the nonlinear mapping ϕ , the proposed approach works well with the given bases. Note that the nominal MPC controller failed to stabilize the system above friction values of 0.1, so again, we demonstrate that we can improve MPC performance substantially with just a few training samples by combining analytical gradient information and data sampled from the true dynamics.

244 5.5 Model Prediction Error vs. Controller Performance

Much of the previous literature on model learning focuses on open-loop dynamics prediction error. While intuitive, we argue that this is a poor metric when the end goal is closed-loop control performance. In Figure 6a we show that decreasing confidence in the analytical model (by increasing α) increases open-loop dynamics prediction error significantly while having minimal impact on closed loop performance below $\alpha = 0.7$. We found we can often quickly find models "good enough" for

control with just a few training trajectories (typically with a higher value of α), that predicted the 250 open-loop dynamics very poorly. For example, in Fig. 6a at the extremes of $\alpha = 0$ (EDMD) and 251 $\alpha > 0.8$, the open-loop predictions were unstable and diverged, while the closed-loop system still 252 successfully tracked the reference trajectory. This also extends to the MLP example, where MPC 253 tracking performance does not correlate to minimizing loss in the training and test process as seen 254 in Fig. 4. In addition, JDMD matches the Jacobians of that of the nominal model (which has some 255 Jacobian error from the true model), while EDMD has significant Jacobian error as shown in Fig. 256 6b. This further demonstrates the importance of Jacobians over open-loop dynamics prediction in a 257 closed-loop control setting, which may be unsurprising due to the presence of the Jacobians in the 258 feedback-policy of closed-loop controllers. 259

260 6 Limitations

Many of the limitations of the proposed approach derive from the limitations of Koopman ap-261 proaches more broadly. Foremost among these is the sensitivity of performance to the selections 262 of the nonlinear mapping and respective unlifting operation; the current study has not investigated 263 the incorporation of the proposed method in methods which jointly learn both the model and the 264 nonlinear mapping, although the extension should be fairly straightforward. In addition, the bilinear 265 Koopman model assumes the original, nonlinear dynamics to be control-affine, limiting its applica-266 tion to broad dynamical systems in general. Another significant limitation of the current work is lack 267 of demonstration on hardware, something we plan to remedy in the future. Better, in-depth compar-268 isons of the given approach to other approaches beyond a simple MLP would also be enlightening, 269 which were left out due to scope limitations. Additionally, while the presented single rigid-body 270 systems such as a quadrotor or airplane have similar dimensionality to many autonomous systems 271 of interest, extensions to systems with many degrees of freedom may be difficult computationally, 272 given derivative information grows with the square of the state dimension. In addition, the relation-273 ship between closed-loop performance and open-loop dynamics prediction error should be studied 274 futher, given we have demonstrated good MPC performance that has not translated directly to model 275 prediction error. As with most data-driven techniques, it is difficult to claim that our method will 276 increase performance in all cases. It is possible that having an extremely poor prior model may 277 hurt rather than help the training process. However, we found that even when the α parameter is ex-278 tremely small (placing little weight on the Jacobians during the learning process), it still dramatically 279 improves the sample efficiency over standard EDMD. It is also quite possible that the performance 280 gaps between EDMD and JDMD shown here can be reduced through better selection of basis func-281 tions and better training data sets; however, given that the proposed approach converges to EDMD 282 as $\alpha \to 0$, we see no reason to not adopt the proposed methodology and simply tune α based on the 283 confidence of the model and the quantity (and quality) of training data. 284

285 7 Conclusions and Future Work

We have presented JDMD, a simple but powerful extension to EDMD that incorporates derivative 286 information from an approximate prior model. We have tested JDMD in combination with a simple 287 linear MPC control policy across a range of systems and tasks, and have found that the resulting 288 combination can dramatically increase sample efficiency over EDMD, often improving over a nom-289 inal MPC policy with just a few sample trajectories. We also showed that the proposed approach 290 is more efficient than a simple multi-layer perception by one or two orders of magnitude. Substan-291 tial areas for future work remain: most notably, demonstrating the proposed pipeline on hardware. 292 Additional directions include applications on sytems with many degrees of freedom such as those 293 whose dynamics are governed by discretized PDEs, lifelong learning or adaptive control applica-294 tions, combining simulated and real data through the use of modern differentiable physics engines 295 [9, 8], residual dynamics learning, as well as the development of specialized numerical methods for 296 solving nonlinear optimal control problems using the learned bilinear dynamics. 297

298 **References**

- [1] F. Farshidian, M. Neunert, A. W. Winkler, G. Rey, and J. Buchli. An efficient optimal planning
 and control framework for quadrupedal locomotion. In 2017 {IEEE} International Conference
 on Robotics and Automation ({ICRA}), pages 93–100. doi:10.1109/ICRA.2017.7989016.
- [2] S. Kuindersma, F. Permenter, and R. Tedrake. An efficiently solvable quadratic program for
 stabilizing dynamic locomotion. pages 2589–2594. ISSN 9781479936854. doi:10.1109/ICRA.
 2014.6907230.
- [3] M. Bjelonic, R. Grandia, O. Harley, C. Galliard, S. Zimmermann, and M. Hutter. Whole-Body
 MPC and Online Gait Sequence Generation for Wheeled-Legged Robots. pages 8388–8395.
 ISSN 9781665417143. doi:10.1109/IROS51168.2021.9636371.
- [4] J. K. Subosits and J. C. Gerdes. From the racetrack to the road: Real-time trajectory replanning
 for autonomous driving. 4(2):309–320. doi:10.1109/TIV.2019.2904390.
- [5] N. Karnchanachari, M. I. Valls, S. David Hoeller, and M. Hutter. Practical Reinforce ment Learning For MPC: Learning from sparse objectives in under an hour on a real robot.
 pages 1-14. doi:10.3929/ETHZ-B-000404690. URL https://doi.org/10.3929/
 ethz-b-000404690.
- [6] D. . Hoeller, F. . Farshidian, M. Hutter, F. Farshidian, and D. Hoeller. Deep Value Model
 Predictive Control. 100:990–1004. doi:10.3929/ETHZ-B-000368961. URL https://doi.
 org/10.3929/ethz-b-000368961.
- [7] Z. Li, X. Cheng, X. B. Peng, P. Abbeel, S. Levine, G. Berseth, and K. Sreenath. Reinforcement
 Learning for Robust Parameterized Locomotion Control of Bipedal Robots. 2021-May:2811–
 2817. ISSN 9781728190778. doi:10.1109/ICRA48506.2021.9560769.
- [8] T. A. Howell, S. L. Cleac'h, J. Z. Kolter, M. Schwager, and Z. Manchester. Dojo: A differentiable simulator for robotics. *arXiv preprint arXiv:2203.00806*, 2022.
- [9] E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In
 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 5026–
 5033. IEEE, 2012. doi:10.1109/IROS.2012.6386109.
- [10] A. Meduri, P. Shah, J. Viereck, M. Khadiv, I. Havoutis, and L. Righetti. BiConMP: A Nonlinear
 Model Predictive Control Framework for Whole Body Motion Planning. doi:10.48550/arxiv.
 2201.07601. URL https://arxiv.org/abs/2201.07601v1.
- [11] D. Bruder, X. Fu, and R. Vasudevan. Advantages of Bilinear Koopman Realizations for the
 Modeling and Control of Systems with Unknown Dynamics. 6(3):4369–4376. doi:10.1109/
 LRA.2021.3068117.
- [12] M. Korda and I. Mezić. Linear predictors for nonlinear dynamical systems: Koopman operator
 meets model predictive control. 93:149–160. doi:10.1016/j.automatica.2018.03.046. URL
 https://doi.org/10.1016/j.automatica.2018.03.046.
- [13] C. Folkestad, D. Pastor, and J. W. Burdick. Episodic Koopman Learning of Nonlinear
 Robot Dynamics with Application to Fast Multirotor Landing. pages 9216–9222. ISSN 9781728173955. doi:10.1109/ICRA40945.2020.9197510.
- [14] H. J. Suh and R. Tedrake. The Surprising Effectiveness of Linear Models for Visual Foresight
 in Object Pile Manipulation. 17:347–363. doi:10.48550/arxiv.2002.09093. URL https:
 //arxiv.org/abs/2002.09093v3.

- [15] C. Folkestad and J. W. Burdick. Koopman NMPC: Koopman-based Learning and Nonlinear
 Model Predictive Control of Control-affine Systems. In *Proceedings IEEE International Conference on Robotics and Automation*, volume 2021-May, pages 7350–7356. Institute of
 Electrical and Electronics Engineers Inc. ISBN 978-1-72819-077-8. doi:10.1109/ICRA48506.
 2021.9562002.
- [16] C. Folkestad, S. X. Wei, and J. W. Burdick. Quadrotor Trajectory Tracking with Learned
 Dynamics: Joint Koopman-based Learning of System Models and Function Dictionaries. URL
 http://arxiv.org/abs/2110.10341.
- [17] A. Narasingam, J. Sang, and I. Kwon. Data-driven feedback stabilization of nonlinear systems:
 Koopman-based model predictive control. pages 1–12.
- [18] SINDy with Control: A Tutorial. URL https://github.com/urban-fasel/SEIR.
- [19] J. L. Proctor, S. L. Brunton, and J. Nathan Kutz. Generalizing koopman theory to allow for
 inputs and control. 17(1):909–930. doi:10.1137/16M1062296. URL http://www.siam.
 org/journals/siads/17-1/M106229.html.
- [20] M. O. Williams, I. G. Kevrekidis, and C. W. Rowley. A Data-Driven Approximation of the Koopman Operator: Extending Dynamic Mode Decomposition. 25(6):1307-1346. doi:10.1007/S00332-015-9258-5/FIGURES/14. URL https://link.springer.com/ article/10.1007/s00332-015-9258-5.
- [21] A. Surana. Koopman Operator Based Observer Synthesis for Control-Affine Nonlinear Systems; Koopman Operator Based Observer Synthesis for Control-Affine Nonlinear Systems.
 ISBN 978-1-5090-1837-6. doi:10.1109/CDC.2016.7799268.
- [22] C. Folkestad, D. Pastor, I. Mezic, R. Mohr, M. Fonoberova, and J. Burdick. Extended Dynamic
 Mode Decomposition with Learned Koopman Eigenfunctions for Prediction and Control. In
 2020 American Control Conference (ACC), pages 3906–3913. IEEE, 2020.
- [23] C. Folkestad, S. X. Wei, and J. W. Burdick. Koopnet: Joint learning of koopman bilinear
 models and function dictionaries with application to quadrotor trajectory tracking. In 2022
 International Conference on Robotics and Automation (ICRA), pages 1344–1350. IEEE, 2022.
- Q. Li, F. Dietrich, E. M. Bollt, and I. G. Kevrekidis. Extended dynamic mode decomposition with dictionary learning: A data-driven adaptive spectral decomposition of the koopman operator. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 27(10):103111, 2017.
- [25] G. Mamakoukas, M. Castano, X. Tan, and T. Murphey. Local Koopman operators for datadriven control of robotic systems. In *Robotics: science and systems*, 2019.
- B. Huang, X. Ma, and U. Vaidya. Feedback stabilization using koopman operator. In 2018
 IEEE Conference on Decision and Control (CDC), pages 6434–6439. IEEE, 2018.
- [27] X. Ma, B. Huang, and U. Vaidya. Optimal quadratic regulation of nonlinear system using
 koopman operator. In *2019 American Control Conference (ACC)*, pages 4911–4916. IEEE,
 2019.
- [28] R. Wang, Y. Han, and U. Vaidya. Deep koopman data-driven optimal control framework for
 autonomous racing. *Early Access*, 5, 2021.
- [29] E. Kaiser, J. N. Kutz, and S. L. Brunton. Data-driven discovery of koopman eigenfunctions for
 control. *Machine Learning: Science and Technology*, 2(3):035023, 2021.
- [30] J. N. Kutz, S. L. Brunton, B. W. Brunton, and J. L. Proctor. *Dynamic mode decomposition: data-driven modeling of complex systems*. SIAM, 2016.

- [31] D. C.-L. Fong and M. Saunders. LSMR: An Iterative Algorithm for Sparse Least-Squares
 Problems. 33(5):2950-2971. ISSN 1064-8275. doi:10.1137/10079687X. URL https:
 //epubs.siam.org/doi/abs/10.1137/10079687X.
- [32] C. C. Paige and M. A. Saunders. LSQR: An Algorithm for Sparse Linear Equations and Sparse Least Squares. 8(1):43-71. ISSN 0098-3500, 1557-7295. doi:10.1145/355984.355989. URL https://dl.acm.org/doi/10.1145/355984.355989.
- [33] P. Strobach. Recursive Least-Squares Using the QR Decomposition. In P. Strobach, editor, *Linear Prediction Theory: A Mathematical Basis for Adaptive Systems*, Springer Series in Information Sciences, pages 63–101. Springer. ISBN 978-3-642-75206-3. doi:10.1007/978-3-642-75206-3_4.
 978-3-642-75206-3_4. URL https://doi.org/10.1007/978-3-642-75206-3_4.
- [34] A. Sayed and T. Kailath. *Recursive Least-Squares Adaptive Filters*, volume 20094251 of *Electrical Engineering Handbook*, pages 1–40. CRC Press. ISBN 978-1-4200-4606-9 978-1-4200-4607-6. doi:10.1201/9781420046076-c21. URL http://www.crcnetbase.com/doi/abs/10.1201/9781420046076-c21.
- [35] A. Ghirnikar and S. Alexander. Stable recursive least squares filtering using an inverse QR
 decomposition. In *International Conference on Acoustics, Speech, and Signal Processing*,
 pages 1623–1626 vol.3. doi:10.1109/ICASSP.1990.115736.
- [36] T. A. Howell, B. E. Jackson, and Z. Manchester. ALTRO: A Fast Solver for Constrained
 Trajectory Optimization. pages 7674–7679. ISSN 9781728140049. doi:10.1109/IROS40897.
 2019.8967788.
- [37] B. E. Jackson, T. Punnoose, D. Neamati, K. Tracy, R. Jitosho, and Z. Manchester. ALTROC: A Fast Solver for Conic Model-Predictive Control; ALTRO-C: A Fast Solver for Conic
 Model-Predictive Control. ISSN 9781728190778. doi:10.1109/ICRA48506.2021.9561438.
 URL https://github.com/.
- [38] J. Moore, R. Cory, and R. Tedrake. Robust post-stall perching with a simple fixed-wing glider using LQR-Trees. *Bioinspiration & amp\$\mathsemicolon\$ Biomimetics*, 9(2):025013, May 2014. doi:10.1088/1748-3182/9/2/025013. URL https://doi.org/10.1088/
 1748-3182/9/2/025013. Publisher: IOP Publishing.
- [39] Z. Manchester, J. Lipton, R. Wood, and S. Kuindersma. A Variable Forward-Sweep Wing
 Design for Enhanced Perching in Micro Aerial Vehicles. In *AIAA Aerospace Sciences Meeting*.
 URL https://rexlab.stanford.edu/papers/Morphing_Wing.pdf.