Online Learning of Dynamical Systems Using Low-Rank Updates to Physics-Informed Kernel Distribution Embeddings

Kendric Ortiz, *Student Member, IEEE,* Rachel DiPirro, *Student Member, IEEE,*
Adam J. Thorpe, *Member, IEEE,* Meeko Oishi, *Senior Member, IEEE*

***Abstract*— In stochastic and dynamic environments, the ability to infer an accurate model of the underlying dynamical system is crucial for ensuring objectives such as responsiveness, performance, or reliability. We present a novel approach to update predictive models of discrete-time, stochastic, dynamical systems in an online fashion. Our approach is based in physics-informed conditional distribution embeddings, a nonparametric machine learning technique that approximates an integral operator to assess the most likely distribution. We propose an efficient numerical method to update the predictive model as new data is gathered, employing low-rank updates. We validate our approach on examples of varying complexity, including an F-16 ground collision avoidance scenario.**

I. Introduction

For autonomous systems to operate in complex, dynamic, real-world environments, it is important for them to be responsive to events and effects that can alter the underlying dynamics and stochasticity. External disturbances (i.e., wind gusts), system failures (i.e., faulty sensors, engine failure, or other malfunctioning elements), and other adverse events (misclassification in perception or control, unexpected human input or interaction) can all impact the accuracy of *a priori* dynamical system models. Mathematical models can capture a great deal of autonomous system behaviors, but may be suspect in environments with significant disturbances. Physics-informed learning approaches, in which data-driven methods augment mathematical models, can provide responsivity to changes in the dynamics or the environment, without ignoring the important knowledge gained from mathematical models. However, most of these approaches presume data gathered a priori, and are not responsive to the needs of near run-time implementation.

Online dynamical learning frameworks seek to address this problem through a variety of approaches. The Koopman operator uses a recursive variant of the canonical dynamic mode decomposition [1, 2], and has shown promise in online system identification and control of dynamical systems [3, 4]. Online Gaussian Process (GP) methods have been explored for their adaptive capabilities in dynamic environments, including UAV systems and robotic applications [5]–[8]. We choose to utilize kernel distribution embeddings for their ability to efficiently incorporate online updates, lack of assumptions on the underlying distribution, and nonparametric flexibility.

In this paper, we propose a method for near run-time updates to physics-informed kernel embeddings. Kernel embeddings of distributions are a class of nonparametric machine learning techniques that allow the reprsentation of integral operators and computation of expectations as inner products in a high-dimensional space of functions known as a reproducing kernel Hilbert space (RKHS). Originally presented in [9, 10], kernel embeddings have been shown to be useful for solving approximate reformulations of stochastic optimal control problems, including dynamic programming and chance-constrained control [11]–[13]. These techniques have also been applied to Markov models [14]–[16], state filtering and estimation [10, 17], and policy synthesis [11, 12, 18].

Our main contribution is a method that enables online updates to data-driven predictive models via kernel embeddings of distributions through low-rank updates to the kernel embedding. Using kernel distribution embeddings to learn the stochastic kernel that describes the system dynamics and uncertainty, we develop an efficient update method for the kernel embedding by exploiting the structure of the empirical kernel embedding estimate that can be computed as low rank matrix updates. We presume a moving window of observations, and seek efficient computational methods to update the embedding that exploits the similarity of the data at each time step. Our approach involves use of rank-one updates to account for adding the most recent sample, and removing the oldest sample. Our approach functions similarly to those based in Cholesky decomposition and QR decomposition, broadening the suite of numerical methods that could be employed for run-time updates for kernel embeddings.

The rest of the paper is organized as follows: In section II we present the mathematical preliminaries on our system model and physics informed kernel embeddings, followed by our problem formulation. In section III we provide our proposed approach and algorithms for the addition and removal of data of the kernel based predictive model. In Section IV we provide a comprehensive analysis of accuracy, scalability, and computational efficiency, followed by a practical application of our method on an F-16 aircraft model. We

This material is based upon work supported by the National Science Foundation under NSF Grant Numbers CNS-1836900 and CMMI-2227338. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. The NASA University Leadership initiative (Grant #80NSSC20M0163) provided funds to assist the authors with their research, but this article solely reflects the opinions and conclusions of its authors and not any NASA entity.

K. R. Ortiz, R. DiPirro, and M. Oishi are with Electrical & Computer Eng., University of New Mexico, Abq., NM. A. Thorpe is with the Oden Institute for Computational Engineering and Sciences, at the University of Texas at Austin, TX, USA. Email: {kendric,rdipirro,ajthor,oishi}@unm.edu, adam.thorpe@austin.utexas.edu.

provide some concluding remarks and future work directions in Section V.

II. Preliminaries & Problem Formulation

Notation: We denote the sets of real numbers as $\mathbb{R}$, and natural numbers as $\mathbb{N}$. Given a space E and $N \in \mathbb{N}$ we denote the Cartesian product $E^N \triangleq E \times \cdots \times E$ (N times). We denote the Borel σ-algebra on a topological space X by $\mathscr{B}_X$, and the expectation operator with respect to Q as $\mathbb{E}_Q$.

A. System Model

Let $\mathcal{X} \subseteq \mathbb{R}^n$ be the state space of the system

$$x_{t+1} = f(x_t, w_t), \tag{1}$$

where $x_t \in \mathcal{X}$ is the state at time t, and $\omega = (\omega_t)_{t\in\mathbb{N}}$ is a stochastic process characterizing a disturbance on the system. The system (1) evolves from an initial condition $x_0 \in \mathcal{X}$, which may be drawn from an initial distribution $\mathbb{P}_0$ over a finite time horizon $t = 0, 1, \ldots, N$ for $N \in \mathbb{N}$. As shown in [19], the system dynamics in (1) can equivalently be represented using a stochastic kernel $Q : \mathscr{B}_\mathcal{X} \times \mathcal{X} \to [0, 1]$, which is a Borel-measurable function that maps a probability measure $Q(\cdot \mid x)$ to every $x \in \mathcal{X}$ on the measurable space $(\mathcal{X}, \mathscr{B}_\mathcal{X})$.

We presume that the system dynamics (1) are *unknown*, meaning that we do not have direct knowledge of the dynamics themselves nor their uncertainty. Consequently, the stochastic kernel Q is likewise unknown. However, we assume that we do have knowledge of an approximation of the dynamics,

$$\bar{x}_{t+1} = \tilde{f}(\bar{x}_t). \tag{2}$$

and knowledge of the last M observations of (1), in which an observation consists of pairs that describe the state at a given time, and the observed state at the next time step. That is, we define the sample set $\mathcal{S}_t = \{(x_i, y_i)\}_{i=t-M-1}^{t-1}$, with $y_i \sim Q(\cdot|x_{i-1})$ as the set consisting of M observations. We distinguish these M samples over this time horizon with the sample gathered at the $(t+1)^{\text{th}}$ time step, that is, $s_t = (x_t, y_t)$ for $y_t \sim Q(\cdot \mid x_t)$. For ease of notation, we define the set with previous and current observations (i.e., with $M+1$ observations in total), as

$$\mathcal{S}_t^+ = \{(x_{t-M-1}, y_{t-M-1}), \ldots, (x_{t-1}, y_{t-1}), (x_t, y_t)\}. \tag{3}$$

These sets are depicted for clarity in Fig. 1.

B. Physics-Informed Kernel Embeddings for State Prediction

Let $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a symmetric, positive definite kernel function on $\mathcal{X}$. According to the Moore-Aronszajn theorem, there exists a corresponding reproducing kernel Hilbert space $\mathscr{H}$ consisting of functions from $\mathcal{X}$ to $\mathbb{R}$ that has the following properties:

1) for every $x \in \mathcal{X}$, $k(x, \cdot) \in \mathscr{H}$, and
2) for every $x \in \mathcal{X}$ and $g \in \mathscr{H}$, $g(x) = \langle g, k(x, \cdot)\rangle_\mathscr{H}$, which is known as the reproducing property.

The reproducing property is key to our approach, since it allows us to evaluate any function in the RKHS as an inner product. As shown in [10], given a probability measure $Q(\cdot \mid x)$, we can represent the expectation operator with respect to $Q(\cdot \mid x)$ as an element $\mu : \mathcal{X} \to \mathscr{H}$ in an RKHS, known as the kernel distribution embedding, such that for any function $g \in \mathscr{H}$, the expectation of g with respect to $Q(\cdot \mid x)$ can be computed via the reproducing property,

$$\mathbb{E}_{Q(\cdot|x)}[g(x')] = \langle g, \mu(x)\rangle_\mathscr{H}. \tag{4}$$

However, in practice since the stochatic kernel Q is unknown, we do not have access to the kernel mean embedding μ directly. Instead, we can compute an empirical estimate $\hat{\mu}$ of μ using the collected data sample $\mathcal{S}_t$. As in shown [20], the empirical estimate $\hat{\mu}$ can be computed as the solution to a regularized least-squares problem that includes an additional bias term in order to account for the approximate knowledge of the system dynamics in (2). Given a sample $\mathcal{S}$ and an estimate $\tilde{f}$ as in (2), the biased regularized least-squares problem is given by

$$\min_{g\in\mathcal{V}} \frac{1}{2\lambda}\sum_{i=1}^{M} \|k(y_i, \cdot) - g(x_i)\|_\mathscr{H}^2 + \frac{1}{2}\|g\|_\mathcal{V}^2 - \langle g, g_0\rangle_\mathscr{H}, \tag{5}$$

where x_i denotes the i^{th} state of the system and $y_i \sim Q(\cdot|x_{i-1})$, $\lambda > 0$ is a regularization parameter that is strictly positive to ensure the problem is well-posed, $\mathcal{V}$ is a vector-valued RKHS consisting of functions from $\mathcal{X}$ to $\mathscr{H}$, and $g_0(x) = k(\tilde{f}(x), \cdot)$ is a bias term that encodes prior knowledge of the dynamics such that for any $g \in \mathscr{H}$,

$$\langle g, g_0(x)\rangle_\mathscr{H} = \langle g, k(\tilde{f}(x), \cdot)\rangle_\mathscr{H} = g(\tilde{f}(x)). \tag{6}$$

As shown in [20], the problem in (5) admits a closed-form solution, given by

$$\hat{\mu}(x) = (\Phi^\top - \tilde{\Phi}^\top)WK(x) + k(\tilde{f}(x), \cdot), \tag{7}$$

where Φ and $\tilde{\Phi}$ are feature vectors, with elements $\Phi_i = k(y_i, \cdot)$ and $\tilde{\Phi}_i = k(\tilde{f}(x_i), \cdot)$, $W = (G + \lambda I)^{-1}$, where $G \in \mathbb{R}^{M\times M}$ is a matrix with elements $G_{ij} = k(x_i, x_j)$, and $K(x) \in \mathbb{R}^M$ is a vector that depends on x with elements $K_i(x) = k(x_i, x)$. As discussed in [20], the empirical estimate $\hat{\mu}$ in (7) has a simple intuitive explanation. The estimate $\hat{\mu}$ consists of two terms: a data-driven part $(\Phi^\top - \tilde{\Phi}^\top)WK(x)$ that captures the difference between

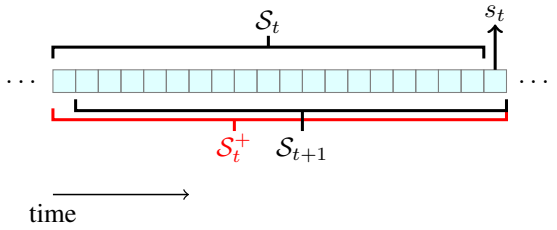

Fig. 1: The data S_t contains M samples, gathered from time $t-M-1$ to time $t-1$. Upon observation of the sample s_t at time step t, we construct $S_t^+ = S_t \cup s_t$, then we construct $S_{t+1} = S_t^+ \backslash s_{t-M-1}$ as the M samples associated with time step $t+1$.

the approximate dynamics and the data, and a correction term $k(\tilde{f}(x), \cdot)$ that shifts the learned function such that it is centered around the bias.

The solution to (7) scales with dimension $O(n^3)$ which is excessive for near run-time implementations.

C. Problem Formulation

We seek a computationally efficient approach to solving (7), that exploits the fact that with a moving horizon of samples, many of the encoded data points already exist in the Hilbert space.

Specifically we seek to solve the following:

Problem 1. *Given an embedding $\hat{\mu}_t$, which is an evaluation of (7) based on a set of M prior samples $\mathcal{S}_t$ and on the approximate dynamics (2), as well as the latest observation s_t, we seek to compute*

$$\mathbb{E}_{Q(\cdot|x_t)}[x_{t+1}], \tag{8}$$

in a computationally efficient manner that exploits the intersection between samples S_t and S_{t+1}, i.e., the overlap due to the moving horizon.

We seek to avoid the direct computation of a matrix inversion at each time step. There are a variety of solutions to Problem 1 that employ methods in linear solvers, such as Cholesky decomposition [21] or QR methods [22, 23]. We propose here an alternative to these approaches, based in block matrix partitions, because it exploits known structure. Numerical solvers may be tailored to particular scenarios, and so the approach we provide is complementary to those based in Cholesky decomposition or QR approaches. We propose a method that efficiently computes the inverse in (7) for the embedding at time $t+1$ by exploiting its structural similarity to the embedding at time t.

Our method efficiently updates the kernel embedding (7) by incorporating new data points and removing the oldest ones, utilizing linear algebra techniques such as the Woodbury matrix identity and block partitioned matrix inversion lemmas, providing the same computational efficiency as a Cholesky decomposition based approach.

III. Methods

Our approach is to sequentially update the embedding by first accounting for the new sample, that is, by calculating the embedding based on sample set S_t^+, then by accounting for removing the oldest sample, which results in calculating the embedding based on sample set S_{t+1}.

A. Adding an Observation

We presume that we have previously computed the embedding (7) using the sample set $\mathcal{S}_t$. In this subsection, we seek to update the embedding (7) with the sample set S_t^+, by exploiting what we already have done to compute S_t.

We define the embedding associated with set S_t^+ as

$$\hat{\mu}_+(x) = (\Phi_+^\top - \tilde{\Phi}_+^\top) W_+ K_+(x) + k(\tilde{f}(x), \cdot). \tag{9}$$

We seek to compute (9) efficiently, by exploiting the solution to (7).

We note that we can use matrix decomposition to rewrite the elements of (9) in terms of the elements of (7), that is, of the embedding associated with the set S_t. Specifically, the feature vectors Φ_+ and $\tilde{\Phi}_+$ can be written as

$$\Phi_+ = \begin{bmatrix} \Phi \\ k(y_{M+1}, \cdot) \end{bmatrix}, \quad \tilde{\Phi}_+ = \begin{bmatrix} \tilde{\Phi} \\ k(\tilde{f}(x_{M+1}), \cdot) \end{bmatrix}. \tag{10}$$

We also note the vector $K_+(x) \in \mathbb{R}^{M+1}$ and the matrix $W_+ \in \mathbb{R}^{(M+1)\times(M+1)}$ can also be decomposed, as

$$K_+(x) = \begin{bmatrix} K(x) \\ k(x_{M+1}, x) \end{bmatrix} \tag{11}$$

$$W_+ = \left(\begin{bmatrix} G & K(x_{M+1}) \\ K(x_{M+1})^\top & k(x_{M+1}, x_{M+1}) \end{bmatrix} + \lambda I \right)^{-1}. \tag{12}$$

The matrix W_+ is symmetric and positive semi-definite (as is W). However, computing the inverse in (12) is the main difficulty to real-time computation. We seek an alternative to compute (12), that takes advantage of our knowledge of W, $K(x)$, and Φ, and employs only matrix multiplication and addition.

Proposition 1 (Matrix inversion lemma, [24], Prop. 2.8.7)**.** *Consider a partitioned matrix $A = \left[\begin{smallmatrix} A_{12} & A_{12} \\ A_{21} & A_{22} \end{smallmatrix}\right] \in \mathbb{R}^{n\times n}$, with block diagonal elements $A_{11} \in \mathbb{R}^{n_1 \times n_1}$ and $A_{22} \in \mathbb{R}^{n_2 \times n_2}$ of dimensions $n_1 + n_2 = n$, that are invertible, then the partition block matrix inversion of A can be written as:*

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}^{-1} = \begin{bmatrix} F & 0 \\ 0 & H \end{bmatrix} \begin{bmatrix} I & -A_{12}A_{22}^{-1} \\ -A_{21}A_{11}^{-1} & I \end{bmatrix} \tag{13}$$

where $F = (A_{11} - A_{12}A_{22}^{-1}A_{21})^{-1}$ and $H = (A_{22} - A_{21}A_{11}^{-1}A_{12})^{-1}$.

The factored form in (13) requires inversion of A_{11} and A_{22}, the expression in H, and the expression in F.

Lemma 1 (Woodbury matrix identity [25])**.** *Let B be an invertible matrix, and let U, C, and V be conformable matrices, meaning the dimensions are suitable for the operation given. Then,*

$$(B_1 + UB_2V)^{-1} = B_1^{-1} - B_1^{-1}U(B_2^{-1} + VB_1^{-1}U)^{-1}VB_1^{-1} \tag{14}$$

By partitioning W_+ with $A_{11} = G + \lambda I$, $A_{12} = K(x_{M+1})$, $A_{21} = K(x_{M+1})^T$, and $A_{22} = k(x_{M+1}, x_{M+1}) + \lambda$, we see that A_{22} and H are scalar, i.e., $n_1 = M, n_2 = 1$ for $n = M+1$. Further, using these substitutions along with Lemma 1 with $B_1 = W^{-1}$, $U = -K(x_{M+1})$, $B_2 = (k + \lambda I)^{-1}$, $V = K(x_{M+1})^T$, we can simplify F as

$$\begin{aligned} F &= \left(W^{-1} - K(k+\lambda)^{-1}K^T\right)^{-1} \\ &= W + WK(k + \lambda - K^TWK)^{-1}K^TW \\ &= W + WKHK^TW \end{aligned} \tag{15}$$

where, with a slight abuse of notation, we use K to indicate $K(x_{M+1})$. Hence by using 15, the solution to $\hat{\mu}_+$ in

(9) is found by computing W_+ using W, without explicitly computing the matrix inverse. We present the algorithm for computing W_+ from W in Algorithm 1.

Algorithm 1 Computing W_+ from W

Input: W, $K(x)$, s_t
Output: updated matrix W_+

$A_{22} \leftarrow k(x_{M+1}, x_{M+1}) + \lambda$
$T \leftarrow K^\top W K$
$H \leftarrow (A_{22} - T)^{-1}$
$F \leftarrow W + WKHK^\top W$
$W_+ \leftarrow \begin{bmatrix} F & 0 \\ 0 & H \end{bmatrix} \begin{bmatrix} I & -KA_{22}^{-1} \\ -K^\top W & I \end{bmatrix}$
Return W_+

B. Removing an Observation

In this section we consider a similar problem, that is, we seek to compute an update to the embedding (7) after removing the oldest observation, without explicitly recomputing the matrix inverse term in (7). We remove the oldest measurement from $\mathcal{S}_t^+$, and compute the embedding associated with $\mathcal{S}_{t+1}$,

$$\hat{\mu}_-(x) = (\Phi_-^\top - \tilde{\Phi}_-^\top) W_- K_-(x) + k(\tilde{f}(x), \cdot). \tag{16}$$

We assume we have access to (9) and its components, Φ_+, K_+, and W_+, and seek to compute W_-. We rewrite the feature vectors Φ_+, vector $K_+(x) \in \mathbb{R}^M$ and the matrix $W_+ \in \mathbb{R}^{M \times M}$ as:

$$\Phi_+ = \begin{bmatrix} \Phi_- \\ k(y_1, \cdot) \end{bmatrix} \tag{17}$$

$$K_+(x) = \begin{bmatrix} K_-(x) \\ k(x_1, x) \end{bmatrix} \tag{18}$$

$$W_+ = \left(\begin{bmatrix} G_- & K(x_1) \\ K(x_1)^\top & k(x_1, x_1) \end{bmatrix} + \lambda I \right)^{-1}. \tag{19}$$

We rewrite W_+ as $(G_+ + \lambda I)^{-1}$ to make explicit the structure within the matrix.

Lemma 2 (Inverse of partitioned matrix, [26])**.** *Given a partitioned matrix $A = \left[\begin{smallmatrix} A_{12} & A_{12} \\ A_{21} & A_{22} \end{smallmatrix}\right] \in \mathbb{R}^{n \times n}$, with block diagonal elements $A_{11} \in \mathbb{R}^{n_1 \times n_1}$ and $A_{22} \in \mathbb{R}^{n_2 \times n_2}$ of dimensions $n_1 + n_2 = n$, that are invertible, the inverse of A can be written as*

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}^{-1} = \begin{bmatrix} F & -FA_{12}A_{22}^{-1} \\ -A_{22}^{-1}A_{21}F & A_{22}^{-1} + A_{22}^{-1}A_{21}FA_{12}A_{22}^{-1} \end{bmatrix} \tag{20}$$

with $F = (A_{11} - A_{12}A_{22}^{-1}A_{21})^{-1} \in \mathbb{R}^{n_1 \times n_1}$.

We partition (19) to employ Lemma 2 with $W_+ = \left[\begin{smallmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{smallmatrix}\right]$. We can then solve for W_- directly.

$$W_- = F^{-1} \tag{21}$$

$$= A_{11} - A_{12}A_{22}^{-1}A_{21} \tag{22}$$

With (22), the solution to $\hat{\mu}_-$ in (16) is readily computable without an inverse. We summarize this process in Algorithm 2.

Algorithm 2 Removing an Observation

Input: matrix W_+
Output: updated matrix W_-

Partition $W_+ \in \mathbb{R}^{M+1 \times M+1}$ as: $W_+ \leftarrow \left[\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right]$, such that $a \in \mathbb{R}^{M \times M}$, $b = c^\top \in \mathbb{R}^M$, and $d \in \mathbb{R}$.
$W_- \leftarrow a - bd^{-1}c$
Return W_-

C. Updating the embedding

We combine the steps in Sec. III-A and Sec. III-B at every time step, as described in Algorithm 3. This algorithm has a computational complexity of $O(n^2)$, whereas calculating (7) directly via inversion incurs a computational complexity of $O(n^3)$.

Algorithm 3 Updating the kernel embedding with a moving horizon sample set

Input: S_t, s_t, W, $K(x)$, μ_t
Output: $\hat{\mu}_{t+1}$

$\mathcal{S}_t^+ \leftarrow \mathcal{S}_t \cup s_t$
Rewrite $K_+(x)$ with observation s_t via (11)
Compute W_+ using Algorithm 1 with W, $K_+(x)$, s_t
$S_{t+1} \leftarrow S_t^+ \backslash s_{t-M-1}$
Compute W_- using Algorithm 2 with W_+
$\hat{\mu}_{t+1} \leftarrow \hat{\mu}_-$ via (16)
Return $\hat{\mu}_{t+1}$

The output of Algorithm 3 provides a computationally efficient solution to update the empirical embedding $\hat{\mu}_{t+1}$, addressing Problem 1 by utilizing low-rank updates to incorporate new data while minimizing computational complexity. Unlike [20], which focuses on incorporating prior system knowledge into kernel embeddings, our approach enables real-time updates, making it well-suited for dynamic environments.

D. Stability & Convergence

We wish to characterize the stability of the algorithm and the conditions for its convergence. We rely on the theory of algorithmic stability, which seeks to derive generalization error bounds for learning algorithms, and provides a means to estimate the *risk*, or generalization error, using a type of sensitivity analysis. Unlike other approaches, such as [27], which seek to determine uniform convergence to the mean, sensitivity analysis seeks to determine how much a variation in the training data can influence the estimate provided by a learning algorithm. In our case, this is particularly useful since the bounds describe the stability of (5) in response to changes in the sample set (such as data being removed or altered).

The *risk*, denoted by $R(\hat{\mu})$, measures the expected loss (error) of the solution $\hat{\mu}$ to the regularized least-squares

problem in (5), and is defined as

$$R(\hat{\mu}) = \int_{\mathcal{X}} \|k(x', \cdot) - \hat{\mu}(x)\|^2_{\mathscr{H}} Q(\mathrm{d}x' \mid x). \tag{23}$$

However, we cannot compute the risk directly since Q is unknown. Thus, we seek to bound the risk by its empirical counterpart. Given a sample $\mathcal{S}$, the *empirical risk*, denoted by $R_{\mathcal{S}}(\hat{\mu})$, also known as the empirical error, measures the actual loss of the learning problem, and is defined as

$$R_{\mathcal{S}}(\hat{\mu}) = \frac{1}{2\lambda} \sum_{i=1}^{M} \|k(x_i', \cdot) - \hat{\mu}(x_i)\|^2_{\mathscr{H}} + \frac{1}{2}\|\hat{\mu}\|^2_{\mathcal{V}} - \langle \hat{\mu}, f_0 \rangle_{\mathcal{V}}. \tag{24}$$

As shown in [28], the (unbiased) regularized least-squares problem (5) is algorithmically stable and admits finite sample bounds. An extension of this to the biased case is trivial and directly follows [28] Theorem 1. We present these bounds in Theorem 3 for this extension.

Theorem 3. *Let k be bounded by $\rho < \infty$. For any $M \geq 1$ and any $\delta \in (0, 1)$, with probability $1 - \delta$, the risk R of the regularized least-squares problem in* (5) *is bounded by*

$$R(\hat{\mu}) \leq R_{\mathcal{S}}(\hat{\mu}) + \frac{\sigma^2 \rho^2}{\lambda M} + \left(\frac{2\sigma^2 \rho^2}{\lambda} + \rho \right) \sqrt{\frac{\log(1/\delta)}{2M}}, \tag{25}$$

where $\sigma > 0$ is a coefficient that depends on the choice of the kernel function k and bounds the algorithm loss function,

$$\begin{aligned} &|\|k(x_1, \cdot) - k(x', \cdot)\|^2 - \|k(x_2, \cdot) - k(x', \cdot)\|^2| \\ &\qquad \leq \sigma \|k(x_1, \cdot) - k(x_2, \cdot)\|^2. \end{aligned} \tag{26}$$

Theorem 3 shows that the empirical estimate $\hat{\mu}$ converges in probability to the true embedding μ as the size of the set of samples increases, and also provides finite sample bound [29] on the empirical estimate. We can ensure boundness of the kernel k on ρ, with proof directly following [30, Theorem 2].

IV. Numerical Results

All experiments were performed in Python 3.8.12 on a 11th Gen Intel i7 processor with 16Gb of RAM. We utilize the stochastic optimal control using kernel methods (SOCKS) toolbox [28] to perform the analysis. The code for this paper is available at `https://github.com/unm-hscl/ortiz-ajthor-dipirro-CDC24`.

A. Double integrator system

In this section, we discuss the accuracy and scalability of the proposed method using an N-dimensional stochastic chain of integrators in a regulation problem using a linear feedback controller.

The dynamics are described by

$$\begin{aligned} x_{t+1} = &\begin{bmatrix} 1 & N_s & \frac{1}{2}N_s^2 & \dots & \frac{1}{(n-1)!}N_s^{n-1} \\ 0 & 1 & N_s & & \\ \vdots & & & \ddots & \vdots \\ 0 & 0 & 0 & \dots & N_s \\ 0 & 0 & 0 & \dots & 1 \end{bmatrix} x_t \\ &+ \begin{bmatrix} \frac{1}{n!}N_s^n & \dots & \frac{1}{2}N_s & N_s \end{bmatrix}^T u_t + \alpha \omega_t \end{aligned} \tag{27}$$

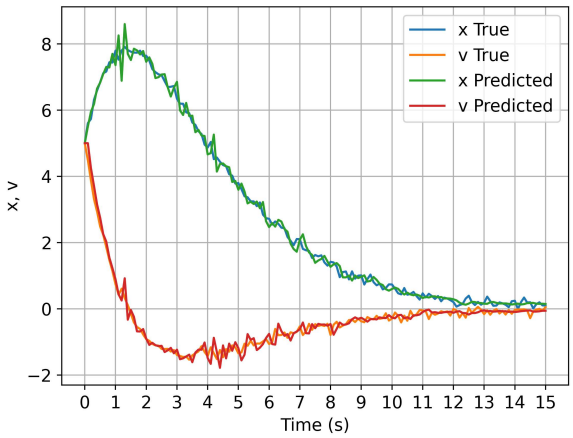

Fig. 2: The proposed method can accurately solve (8) for 2-D integrator dynamics despite a noise scaling factor larger than the step changes in the dynamics.

where $x_t \in \mathbb{R}^n$ is the state, $u_t \in \mathbb{R}$ is the control input, and ω_t is a random variable with distribution $\mathcal{N}(0, 1)$ scaled by $\alpha \in [0.01, 1]$. We define a linear state feedback controller $u = -Kx$ with gain matrix K chosen via pole placement to ensure stability. The sampling time is $N_s = 0.1$ seconds. We solve the single step prediction (8) for (27) using a Guassian kernel function $k(x, x') = \exp(-||x - x'||^2/2\sigma^2)$ with hyperparameters $\sigma = 0.5$ and $\lambda = 1e - 3$.

1) Accuracy: We explore the accuracy of our method using the average mean squared error (AMSE) over ten trials as a function of time, window size γ, and scaled additive noise α. Figure 2 shows the solution to (8) for $n = 2$, over a time horizon of $N = 150$ time steps, with $\alpha = 0.1$. The prediction inaccuracies primarily stem from the large impact of the noise on the dynamics. The solution is numerically identical to the prediction method that uses Cholesky factorization to update W when both approaches are numerically stable. Both the proposed method and the Cholesky decomposition method can suffer from numerical instabilities when the Gram matrix is ill-conditioned [23, §5.3.8]. The Cholesky decomposition can also become numerically unstable when the matrix W^{-1} is near rank deficient [23, §5.3.4].

Figures 3 and 4 depict the average mean square error for the proposed approach as window size and noise scaling vary, respectively. We found empirically that for window sizes less than 10% of the M samples in the moving window, the proposed method suffers from numerical instability, as shown in 3. However, when $\gamma > 0.1M$, we see an increased accuracy of prediction. Figure 4 shows the average 10 trial AMSE of the proposed approach as we scale the noise on the range $[0.01, 1]$. As expected, an increased error is observed as a function of the scaling factor. When sufficient data is collected to characterize the system and the Gram matrix is well-conditioned, there is an increase in the prediction error.

2) Scalability: We evaluate the scalability of the proposed method on (27) for $n \in [2, 50, 100, \dots, 1000]$. As expected,

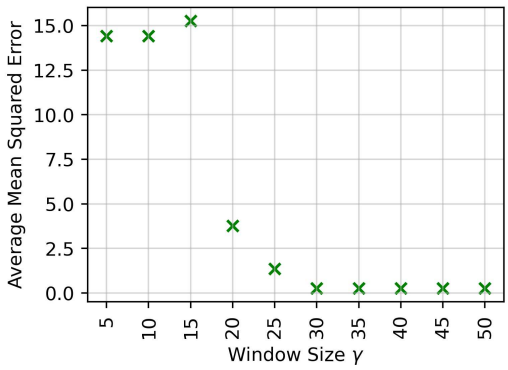

Fig. 3: Average mean square error over 10 trials in the prediction of the states for a 2-D integrator via Algorithm 3, with noise scale $\alpha = 0.1$. Small window sizes lead to poor characterization of the system and can lead to numerical instability with the proposed method. Larger window sizes improve the accuracy of the prediction.

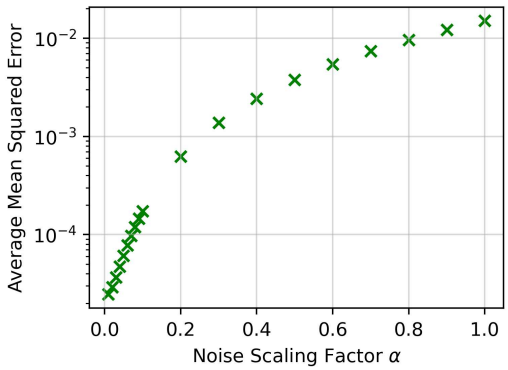

Fig. 4: Average mean square error over 10 trials in the prediction of the states for a 2-D integrator using Algorithm 3 with window size $\gamma = 50$. While the prediction error increases with larger noise scaling, it still remains robust to noise, with magnitude < 0.01.

Figure 5 shows a linear increase in the computation time with dimensionality, associated with the computation time for the state variable predictions.

We also consider computational time as a function of window size. Figure (6) shows the average computation time increases over a window size range $\gamma \in [5, 50]$, up to a window size of approximately 33% of the total data.

B. F-16 Aircraft

In this section we demonstrate our proposed method in a more realistic and complex scenario. We consider a ground collision avoidance scenario for an F-16 aircraft with low altitude and a negative pitch angle [20, 31, 32] , in which the aircraft sequences through multiple controllers to first right the aircraft and then initiate a climb.

We presume that the approximate dynamics (2) are inaccurate, with gravity parameter that is $g = 4.8$ m/s^2. We chose this value to intentionally be a different value than is

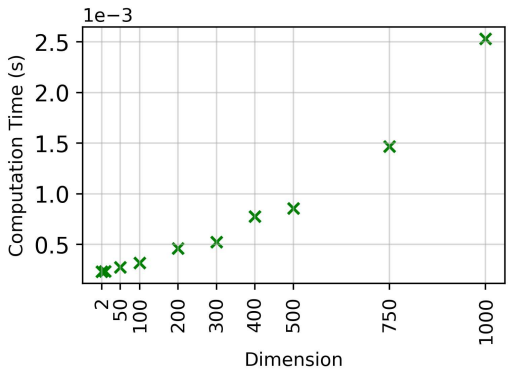

Fig. 5: Computation time for online updates in Algorithm 3 for the n-D integrator with window size $\gamma = 50$ and noise scale $\alpha = 0.1$. The computation time increases linearly due to the additional state elements to be predicted.

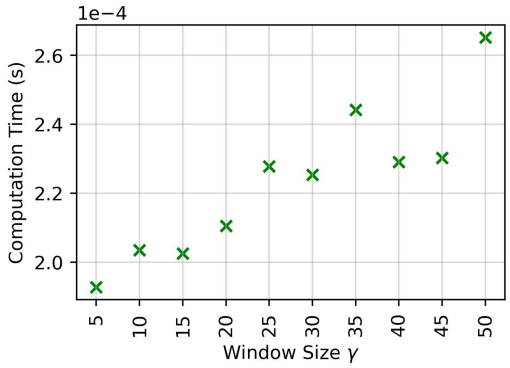

Fig. 6: Average computation time for 10 trials to compute online updates (algorithm 3) over increasing window size γ, with noise scale $\alpha = 0.1$. The computation time increases with increasing window size due to the increased complexity of the matrix computations.

reflected in simulated data, which reflects the true value of $g = 9.8$ m/s^2.

The F-16 dynamics are nonlinear, with 13 states and 4 control inputs. The dynamics capture the 6-DOF motion with a state that consists of velocity v_t, angle of attack α, sideslip β, altitude h, attitude angles roll ϕ, pitch θ, yaw ψ, and their respective rates p, q, r, engine power, and two states, p_n and p_e, which capture translation along north and east, as in [33]. The plant is built on linearly interpolated lookup tables that incorporate wind tunnel data describing the engine model, and other dynamic coefficients. The system is controlled by three independent LQR controllers that switch at unknown times during the ground collision avoidance scenario, posing notable alternations to the closed-loop dynamics during flight.

As was done in [20], we collect an initial data set with 100 samples. We consider a window size of $\gamma = 300$, chosen to be both large enough to sufficiently capture the system dynamics, but also small enough to be responsive to the switch in the dynamics. We use a Gaussian kernel with

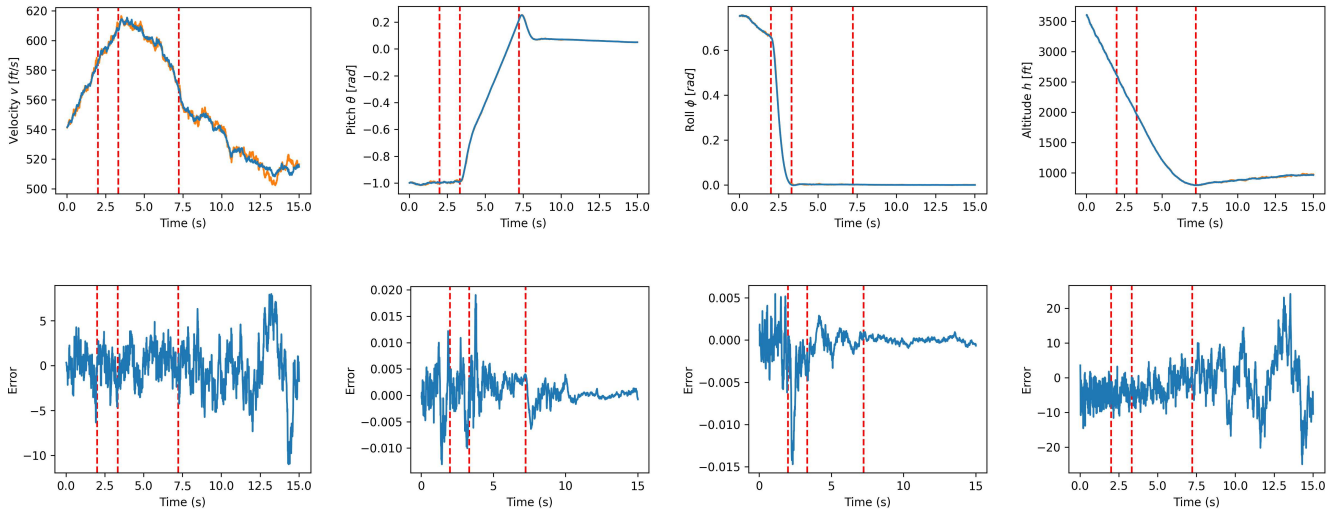

Fig. 7: Upper: Using low-rank updates to physics informed kernel embeddings, we can accurately predict [orange] the true state [blue] of an F-16 performing a ground collision avoidance maneuver, as shown for velocity, pitch, roll, and altitude. Lower: The error between the predicted and true state is low as compared to the overall magnitude of the state.

$\sigma = 0.5$ and a regularization parameter of $\lambda = 1e - 5$.

Figure 7 shows the solution to (8) for state prediction. We can see that the proposed approach can effectively predict the states of the system online. As shown in Figure 7, when the controller switches (shown by the vertical red lines), the state prediction accuracy decreases, because the data no longer characterizes the system behavior due to changes in the underlying controller. However, as time progresses, and the data replaced with samples corresponding to the new dynamics, the prediction rapidly returns to be close to the true dynamics. This importance of this result is that it demonstrates that the proposed approach can accommodate significant changes in the dynamics in near run-time.

V. Conclusion & Future Work

In this paper, we presented a novel technique for incorporating online updates to kernel embeddings for online system identification and prediction. We presented algorithms that enable the addition and removal of data to maintain relevance over time through rank-one updates to the empirical estimate of the embedding. Our approach avoids computationally expensive matrix inversions. We analyzed the accuracy, scalability, and numerical efficiency of our approach on a chain of integrators, and demonstrated its efficacy on a ground collision avoidance scenario for an F-16 aircraft. Our approach is computationally efficient for near run-time applications, and robust to numerical instabilities in the examples considered. Future directions for research includes the development of methods to strategically choose samples online to remove not only the oldest data, but rather obsolete data.

References

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

[2] S. Sinha, S. P. Nandanoori, and E. Yeung, "Online learning of dynamical systems: An operator theoretic approach," 2019. [Online]. Available: https://arxiv.org/abs/1909.12520

[3] H. M. Calderón, E. Schulz, T. Oehlschlägel, and H. Werner, "Koopman operator-based model predictive control with recursive online update," in *2021 European Control Conference (ECC)*, 2021, pp. 1543–1549.

[4] S. Sinha, S. P. Nandanoori, and D. Barajas-Solano, "Online real-time learning of dynamical systems from noisy streaming data," *Scientific Reports*, vol. 13, 12 2023.

[5] D. Nguyen-Tuong and J. Peters, "Incremental online sparsification for model learning in real-time robot control," *Neurocomputing*, vol. 74, no. 11, pp. 1859–1867, 2011, adaptive Incremental Learning in Neural Networks Learning Algorithm and Mathematic Modelling Selected papers from the International Conference on Neural Information Processing 2009 (ICONIP 2009). [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231211000701

[6] L. Wang, E. A. Theodorou, and M. Egerstedt, "Safe learning of quadrotor dynamics using barrier certificates," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 2460–2465.

[7] F. Berkenkamp and A. P. Schoellig, "Safe and robust learning control with gaussian processes," in *2015 European Control Conference (ECC)*, 2015, pp. 2496–2501.

[8] J. Umlauft and S. Hirche, "Feedback linearization based on gaussian processes with event-triggered online learning," *IEEE Transactions on Automatic Control*, vol. 65, no. 10, pp. 4154–4169, 2020.

[9] A. Smola, A. Gretton, L. Song, and B. Schölkopf, "A Hilbert space embedding for distributions," in *International Conference on Algorithmic Learning Theory*. Springer, 2007, pp. 13–31.

[10] L. Song, J. Huang, A. Smola, and K. Fukumizu, "Hilbert space embeddings of conditional distributions with applications to dynamical systems," in *International Conference on Machine Learning*, 2009, pp. 961–968.

[11] A. J. Thorpe and M. M. K. Oishi, "Stochastic optimal control via hilbert space embeddings of distributions," in *2021 60th IEEE Conference on Decision and Control (CDC)*, 2021, pp. 904–911.

[12] A. J. Thorpe, J. A. Gonzales, and M. M. K. Oishi, "Data-driven stochastic optimal control using kernel gradients," 2022.

[13] A. Thorpe, T. Lew, M. Oishi, and M. Pavone, "Data-driven chance constrained control using kernel distribution embeddings,"

in *Proceedings of The 4th Annual Learning for Dynamics and Control Conference*, ser. Proceedings of Machine Learning Research, R. Firoozi, N. Mehr, E. Yel, R. Antonova, J. Bohg, M. Schwager, and M. Kochenderfer, Eds., vol. 168. PMLR, 23–24 Jun 2022, pp. 790–802. [Online]. Available: https://proceedings.mlr.press/v168/thorpe22a.html

[14] S. Grünewälder, G. Lever, B. Ll, M. Pontil, and A. Gretton, "Modelling transition dynamics in MDPs with RKHS embeddings," in *International Conference on Machine Learning*. Omnipress, 2012, pp. 535–542.

[15] L. Song, B. Boots, S. M. Siddiqi, G. Gordon, and A. Smola, "Hilbert space embeddings of hidden markov models," in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ser. ICML'10. Madison, WI, USA: Omnipress, 2010, p. 991–998.

[16] Y. Nishiyama, A. Boularias, A. Gretton, and K. Fukumizu, "Hilbert space embeddings of POMDPs," in *Conference on Uncertainty in Artificial Intelligence*, 2012.

[17] L. Dang, B. Chen, S. Wang, Y. Gu, and J. C. Príncipe, "Kernel kalman filtering with conditional embedding and maximum correntropy criterion," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 66, no. 11, pp. 4265–4277, 2019.

[18] G. Lever and R. Stafford, "Modelling Policies in MDPs in Reproducing Kernel Hilbert Space," in *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, G. Lebanon and S. V. N. Vishwanathan, Eds., vol. 38. San Diego, California, USA: PMLR, 09–12 May 2015, pp. 590–598. [Online]. Available: https://proceedings.mlr.press/v38/lever15.html

[19] D. P. Bertsekas and S. E. Shreve, *Stochastic optimal control: the discrete time case*. Elsevier, 1978.

[20] A. J. Thorpe, C. Neary, F. Djeumou, M. M. K. Oishi, and U. Topcu, "Physics-informed kernel embeddings: Integrating prior system knowledge with data-driven control," 2023.

[21] A.-L. Cholesky, "Sur la résolution numérique des systèmes d'équations linéaires," *Comptes Rendus de l'Académie des Sciences*, vol. 178, pp. 677–680, 1924.

[22] A. S. Householder, *The Theory of Matrices in Numerical Analysis*. Mineola, NY: Dover Publications, 1975.

[23] G. H. Golub and C. F. Van Loan, *Matrix Computations*. Baltimore, MD: Johns Hopkins University Press, 1996.

[24] D. S. Bernstein, *Matrix Mathematics: Theory, Facts, and Formulas*, 2nd ed. Princeton, NJ: Princeton University Press, 2009. [Online]. Available: https://www.stat.uchicago.edu/~lekheng/courses/309/books/Bernstein.pdf

[25] M. A. Woodbury, *Inverting modified matrices*. Statistical Research Group, Princeton University, 1950.

[26] F. R. Gantmacher, *The Theory of Matrices*, ser. Translations of Mathematical Monographs. New York: Chelsea Publishing Co., 1959, vol. 1. [Online]. Available: https://archive.org/details/theoryofmatrices00gant

[27] V. N. Vapnik and A. Y. Chervonenkis, "Necessary and sufficient conditions for the uniform convergence of means to their expectations," *Theory of Probability & Its Applications*, vol. 26, no. 3, pp. 532–553, 1982. [Online]. Available: https://doi.org/10.1137/1126059

[28] A. Thorpe and M. Oishi, "Socks: A stochastic optimal control and reachability toolbox using kernel methods," in *25th ACM International Conference on Hybrid Systems: Computation and Control*, ser. HSCC '22. New York, NY, USA: Association for Computing Machinery, 2022. [Online]. Available: https://doi.org/10.1145/3501710.3519525

[29] A. J. Thorpe, K. R. Ortiz, and M. M. Oishi, "State-based confidence bounds for data-driven stochastic reachability using hilbert space embeddings," *Automatica*, vol. 138, p. 110146, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0005109821006750

[30] O. Bousquet and A. Elisseeff, "Stability and generalization," *Journal of Machine Learning Research*, vol. 2, pp. 499–526, 2002.

[31] F. Djeumou and U. Topcu, "Learning to reach, swim, walk and fly in one trial: Data-driven control with scarce data and side information," in *Proceedings of The 4th Annual Learning for Dynamics and Control Conference*, ser. Proceedings of Machine Learning Research, R. Firoozi, N. Mehr, E. Yel, R. Antonova, J. Bohg, M. Schwager, and M. Kochenderfer, Eds., vol. 168. PMLR, 23–24 Jun 2022, pp. 453–466. [Online]. Available: https://proceedings.mlr.press/v168/djeumou22b.html

[32] P. Heidlauf, A. Collins, M. A. Bolender, and S. Bak, "Verification challenges in f-16 ground collision avoidance and other automated maneuvers," in *ARCH@ADHS*, 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:53082578

[33] B. Stevens, E. Johnson, and F. Lewis, *Aircraft Control and Simulation: Dynamics, Controls Design, and Autonomous Systems*. Wiley, 2016. [Online]. Available: https://books.google.com/books?id=zaFAswEACAAJ