2024 American Control Conference (ACC)
July 8-12, 2024. Toronto, Canada

Memory Sketching for Data-driven Prediction of Dynamical Systems

Minghao Shen[1] and Gábor Orosz[1,2]

***Abstract*— In this paper, we introduce a sketching method for data-driven prediction problems in an online setting. We show that sketching the memory of dynamical systems in a randomized way can achieve a constant time and space complexity in each update. We demonstrate the effectiveness of the proposed sketch-based data-driven prediction on trajectory prediction in vehicular traffic. We show that the sketching method can achieve high prediction accuracy with limited memory space, thus enabling online deployment.**

I. Introduction

Data-driven prediction and control has been attracting increasing attention in recent years, with applications in many fields such as connected and automated vehicles [1], quadcopters [2], and power systems [3]. The approaches can be categorized into *indirect methods* and *direct methods* [4]. Indirect methods use system identification to find a model using input-output data, and then design controllers based on that model. Direct methods skip the system identification step: predictions are made and control actions are synthesized directly from input-output data. In this paper, we show the equivalence between an indirect method and a direct method. We propose an improved direct method, and evaluate it using the indirect method.

With increasing amount of data collected, the size of the associated optimization problem grows, and it becomes computationally expensive and infeasible for online deployment. A common solution is to keep the most recent data, and discard the older ones [5], [6], under the assumption that the latest data are the most relevant to the current system behavior. However, when discarding the older data, one may lose important information about the system and the most recent data may not be sufficient to characterize the system behavior. As a compromise, one can keep the offline training data, and in addition, select the most relevant data for online deployment [7]. In order to extract the most important information from the data, singular value decomposition (SVD) can be used to obtain a minimum dimension representation of the system [8]. However, in the presence of noise, the thresholding of singular values is typically not possible, and performing SVD sequentially is computationally expensive. In addition to the issues mentioned above, the performance of the proposed methods is rarely compared with the case when the full data is utilized.

*This work was supported by the Center for Connected and Automated Transportation, University of Michigan, through the US Department of Transportation (DOT) under Grant 69A3551747105.

[1]Minghao Shen and Gábor Orosz are with the Department of Mechanical Engineering, University of Michigan, Ann Arbor, MI 48109, USA {mhshen, orosz}@umich.edu

[2]Gábor Orosz is also with the Department of Civil and Environmental Engineering, University of Michigan, Ann Arbor, MI 48109, USA.

Our goal is to develop a control framework that uses limited memory such that the prediction accuracy is comparable to the result using the full data. We will show that sketching [9] is a viable tool to solve this problem. Sketching is a popular technique in database technology [10], where data is streaming in and out of the database, and people want to keep track of the most important features (such as the most frequent items) without memorizing the whole database. As a dimension reduction technique, sketching methods are applied in optimization [11], signal processing [12] and machine learning [13].

In this paper, for the first time, we establish sketching tools for dynamical systems and control. Sketching methods have two categories: deterministic and randomized. Deterministic sketching methods, such as frequent directions sketch [14], make use of dimension reduction techniques, such as SVD, but they suffer from intermittent heavy SVD computation. Thus, we focus on randomized sketching based on Johnson-Lindenstrauss transform [15]. The main contributions of the paper are as follows: (i) We propose a randomized *memory sketching* method for online data-driven prediction which can achieve constant space and time complexity for each update. (ii) We provide a quantitative connection between the memory size and the prediction accuracy.

The paper is organized as follows. In Section II, we provide background knowledge on data-driven prediction problems. In Section III, we introduce sketching methods, and then establish the tools for memory sketching when performing data-driven prediction. In Section IV, we demonstrate the effectiveness of the proposed memory sketching in trajectory prediction in vehicular traffic. We conclude the paper and lay out future research directions in Section V.

II. Background on Data-driven Prediction

In this section, we provide some background knowledge on data-driven prediction. Consider a discrete-time LTI system Σ with control input $u \in \mathbb{R}^m$ and system output $y \in \mathbb{R}^p$. Denote $u(k_1 : k_2) = [u(k_1)^\top \; u(k_1+1)^\top \; \cdots \; u(k_2)^\top]^\top$, and define $y(k_1 : k_2)$ similarly. At time k, given the input-output trajectory $u^{\mathrm{p}} = u(k - T^{\mathrm{p}} + 1 : k)$, $y^{\mathrm{p}} = y(k - T^{\mathrm{p}} + 1 : k)$ in the past horizon T^{p} and the control input $u^{\mathrm{f}} = u(k+1 : k+T^{\mathrm{f}})$ in the future horizon T^{f}, the task is to predict the output $y^{\mathrm{f}} = y(k+1 : k+T^{\mathrm{f}})$ over the future horizon T^{f}. In *data-driven prediction*, we have access to a system input-output trajectory $(u^{\mathrm{d}}, y^{\mathrm{d}})$. We will investigate how to utilize the data to predict the future output y^{f}.

As mentioned above, data-driven prediction can be categorized into direct and indirect methods. Direct methods for

discrete-time LTI system are based on Willems' behavioral theory [16]–[18]. Consider the space of system trajectories of length K:

$$\mathcal{W}_K(\Sigma) = \left\{ [u(0:K-1)^\top \quad y(0:K-1)^\top]^\top \right. \\ \left. | \{u, y\} \text{ are trajectories of } \Sigma \right\}, \tag{1}$$

where $u(0:K-1) = [u(0)^\top \cdots u(K-1)^\top]^\top$, and $y(0:K-1)$ is defined similarly. For $k < K$, we can define the Hankel matrix of order k as

$$\mathcal{H}_k(u(0:K-1)) = \begin{bmatrix} u(0) & u(1) & \cdots & u(K-k) \\ u(1) & u(2) & \cdots & u(K-k+1) \\ \vdots & \vdots & \ddots & \vdots \\ u(k-1) & u(k) & \cdots & u(K-1) \end{bmatrix}. \tag{2}$$

Willems' fundamental lemma shows that $\mathcal{W}_K(\Sigma)$ can be fully characterized by the input-output trajectory data $(u^{\mathrm{d}}, y^{\mathrm{d}})$ assuming that the input data u^{d} is rich enough. The richness of the system input is measured as follows.

Definition 1 (Persistence of Excitation)**.** The input trajectory u^{d} is *persistently exciting* of order k if the Hankel matrix $\mathcal{H}_k(u^{\mathrm{d}})$ has full row rank.

This enables us to state the fundamental lemma.

Lemma 1 (Fundamental Lemma [19])**.** *Let $(u^{\mathrm{d}}, y^{\mathrm{d}})$ be an input-output trajectory of discrete-time LTI system Σ. If the input trajectory u^{d} is persistently exciting of order $n + K$, where n is the state dimension of the minimal representation of Σ, then $\mathcal{W}_K(\Sigma)$ can be represented as the range space of the Hankel matrix:*

$$\mathcal{W}_K(\Sigma) = \operatorname{range}\left(\begin{bmatrix} \mathcal{H}_K(u^{\mathrm{d}}) \\ \mathcal{H}_K(y^{\mathrm{d}}) \end{bmatrix} \right). \tag{3}$$

The fundamental lemma enables *data-driven predictions* for LTI systems. Given input-output trajectory $(u^{\mathrm{d}},\ y^{\mathrm{d}})$ of length L, choosing $K = T^{\mathrm{p}} + T^{\mathrm{f}}$, one can obtain the prediction y^{f} by solving the quadratic optimization problem:

$$\min_{g \in \mathbb{R}^N, y^{\mathrm{f}} \in \mathbb{R}^{pT^{\mathrm{f}}}} \frac{1}{2}\|g\|_2^2, \quad \text{s.t.} \begin{bmatrix} \mathcal{H}_{T^{\mathrm{p}}+T^{\mathrm{f}}}(u^{\mathrm{d}}) \\ \mathcal{H}_{T^{\mathrm{p}}+T^{\mathrm{f}}}(y^{\mathrm{d}}) \end{bmatrix} g = \begin{bmatrix} u^{\mathrm{p}} \\ u^{\mathrm{f}} \\ y^{\mathrm{p}} \\ y^{\mathrm{f}} \end{bmatrix}, \tag{4}$$

where $g \in \mathbb{R}^N$ and $N = L - K + 1$, which scales linearly with the length of the input-output trajectory L.

To simplify problem (4), we split the data matrix into two:

$$\mathcal{H}_{T^{\mathrm{p}}+T^{\mathrm{f}}}(u^{\mathrm{d}}) = \begin{bmatrix} U^{\mathrm{p}} \\ U^{\mathrm{f}} \end{bmatrix}, \quad \mathcal{H}_{T^{\mathrm{p}}+T^{\mathrm{f}}}(y^{\mathrm{d}}) = \begin{bmatrix} Y^{\mathrm{p}} \\ Y^{\mathrm{f}} \end{bmatrix}, \tag{5}$$

where $U^{\mathrm{p}} \in \mathbb{R}^{mT^{\mathrm{p}} \times N}$, $U^{\mathrm{f}} \in \mathbb{R}^{mT^{\mathrm{f}} \times N}$, $Y^{\mathrm{p}} \in \mathbb{R}^{pT^{\mathrm{p}} \times N}$, $Y^{\mathrm{f}} \in \mathbb{R}^{pT^{\mathrm{f}} \times N}$. Using the Karush–Kuhn–Tucker (KKT) condition, (4) results in

$$g + \begin{bmatrix} U^{\mathrm{p}\top} & U^{\mathrm{f}\top} & Y^{\mathrm{p}\top} & Y^{\mathrm{f}\top} \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} = 0, \quad \lambda_2 = 0, \tag{6}$$

where $\lambda_1 \in \mathbb{R}^{(m+p)T^{\mathrm{p}}+mT^{\mathrm{f}}}$, $\lambda_2 \in \mathbb{R}^{pT^{\mathrm{f}}}$ are Lagrange multipliers corresponding to $\begin{bmatrix} U^{\mathrm{p}\top} & U^{\mathrm{f}\top} & Y^{\mathrm{p}\top} \end{bmatrix}^\top$, and Y^{f} respectively. Since $\lambda_2 = 0$, we can rewrite the KKT condition as

$$g + \Xi^\top \lambda_1 = 0, \tag{7}$$

where $\Xi = \begin{bmatrix} U^{\mathrm{p}} \\ U^{\mathrm{f}} \\ Y^{\mathrm{p}} \end{bmatrix}$. Thus, defining $\xi = \begin{bmatrix} u^{\mathrm{p}} \\ u^{\mathrm{f}} \\ y^{\mathrm{p}} \end{bmatrix}$, the optimization problem (4) can be solved by first solving the least squares problem

$$\hat{g} = \operatorname*{argmin}_g \frac{1}{2}\|g\|_2^2, \quad \text{s.t. } \Xi g = \xi, \tag{8}$$

and then using equality constraint to predict the future output

$$\hat{y}^{\mathrm{f}} = Y^{\mathrm{f}} \hat{g}. \tag{9}$$

In a direct method of data-driven prediction, we represent ξ as a linear combination of the columns of Ξ, and then use this linear combination to predict the future output. However, such direct method does not provide a model for the system, and thus, the evaluation of the method is not straightforward. Fortunately, we can build the connection the proposed direct method (8),(9) to the indirect method as follows.

Lemma 2. *Given Ξ, Y^{f} and ξ, the estimated $\hat{y}^{\mathrm{f}}$ in (8),(9) is equivalent to*

$$\hat{\Phi} = \operatorname*{argmin}_\Phi \frac{1}{2}\|\Phi\Xi - Y^{\mathrm{f}}\|_{\mathrm{F}}^2, \quad \hat{y}^{\mathrm{f}} = \hat{\Phi}\xi. \tag{10}$$

where $\|\cdot\|_{\mathrm{F}}^2$ denotes the Frobenius norm of matrices.

Proof. Using the Moore-Penrose pseudo-inverse $\dagger$, the solution to (8) is $\hat{g} = \Xi^\dagger \xi$, yielding $\hat{y}^{\mathrm{f}} = Y^{\mathrm{f}}\hat{g} = Y^{\mathrm{f}}\Xi^\dagger\xi$. On the other hand, the solution to (10) is $\hat{\Phi} = Y^{\mathrm{f}}\Xi^\dagger$, yielding $\hat{y}^{\mathrm{f}} = \hat{\Phi}\xi = Y^{\mathrm{f}}\Xi^\dagger\xi$. Thus, (8),(9) are equivalent to (10). ■

Lemma 2 explicitly decomposes the data-driven prediction into system identification and prediction. We directly consider the linear map from Ξ to Y^{f}, which is referred to as *multi-step linear predictor* [4], [20] in the literature. In the following sections, we will evaluate our proposed direct method based on the underlying multi-step linear predictor.

III. Memory Sketching

Data-driven prediction bypasses the system identification step, thus enabling convenient online deployment. As new input-output data is collected, those can be appended to the last column of data matrices Ξ and Y^{f}. If there is no noise in the observation, storing the whole data is not necessary, after the column space of data matrix $[U^{\mathrm{p}\top}\ U^{\mathrm{f}\top}\ Y^{\mathrm{p}\top}\ Y^{\mathrm{f}\top}]^\top$ spans the $\mathcal{W}_K(\Sigma)$. However, with noise in the observation, more data is needed to improve the prediction accuracy and the width of the Hankel matrix N grows linearly with time. In this section, we study data-driven prediction with a fixed memory size.

Consider $\Xi \in \mathbb{R}^{d \times N}$ and $Y^{\mathrm{f}} \in \mathbb{R}^{pT^{\mathrm{f}} \times N}$, where $d = (m+p)T^{\mathrm{p}} + mT^{\mathrm{f}}$. We call the linear map

$$\tilde{\Xi} = \Xi\Pi, \quad \tilde{Y}^{\mathrm{f}} = Y^{\mathrm{f}}\Pi, \tag{11}$$

a *sketch* of Ξ and Y^{f}, where $\Pi \in \mathbb{R}^{N\times S}$ is the *sketch matrix*, and S is the *memory size*. Then in the data-driven prediction (8) (9), we replace Ξ and Y^{f} with $\tilde{\Xi}$ and $\tilde{Y}^{\mathrm{f}}$, respectively:

$$\tilde{g} = \underset{g}{\operatorname{argmin}} \frac{1}{2}\|g\|_2^2, \quad \text{s.t. } \tilde{\Xi} g = \xi, \text{ and } \tilde{y}^{\mathrm{f}} = \tilde{Y}^{\mathrm{f}} \tilde{g}, \quad (12)$$

which can be re-written as

$$\tilde{g} = \underset{g}{\operatorname{argmin}} \frac{1}{2}\|g\|_2^2, \quad \text{s.t. } \begin{bmatrix} \tilde{\Xi} \\ \tilde{Y}^{\mathrm{f}} \end{bmatrix} \Pi g = \begin{bmatrix} \xi \\ \tilde{y}^{\mathrm{f}} \end{bmatrix}. \quad (13)$$

When the data is generated without noise, any feasible solution of (13) gives a valid trajectory of the underlying system due to Lemma 1.

Similar to how (8), (9) was reformulated as (10), now (12) or (13) can be reformulated as

$$\tilde{\Phi} = \underset{\Phi}{\operatorname{argmin}} \frac{1}{2}\|\Phi\tilde{\Xi} - \tilde{Y}^{\mathrm{f}}\|_{\mathrm{F}}^2, \quad \tilde{y}^{\mathrm{f}} = \tilde{\Phi}\xi. \quad (14)$$

This is used to determine to what extent a sketching method captures the system dynamics. Namely, we evaluate the in-sample error of estimator $\tilde{\Phi}$ trained with the sketched data:

$$\tilde{J} = \|\tilde{\Phi}\Xi - Y^{\mathrm{f}}\|_{\mathrm{F}}^2. \quad (15)$$

and compare this to the error of the estimator $\hat{\Phi}$ in (10) trained with the full data:

$$\hat{J} = \|\hat{\Phi}\Xi - Y^{\mathrm{f}}\|_{\mathrm{F}}^2. \quad (16)$$

A straightforward way to reduce the memory size is to keep the most recent data. We call this *window sketch*. Keeping the most recent S data points yields the linear map

$$\tilde{\Xi}^{\mathrm{win}} = \Xi\Pi^{\mathrm{win}}, \quad \tilde{Y}^{\mathrm{f,win}} = Y^{\mathrm{f}}\Pi^{\mathrm{win}}, \quad (17)$$

with

$$\Pi^{\mathrm{win}} = \begin{bmatrix} 0_{(N-S)\times S} \\ I_S \end{bmatrix}, \quad (18)$$

where $0_{(N-S)\times S}$ is zero matrix of size $(N-S)\times S$, and I_S is identity matrix of size $S \times S$. The window sketch is deterministic as the sketch matrix Π^{win} is determined given memory size S. It is also data oblivious since the sketch matrix Π^{win} is independent of the data matrices Ξ and Y^{f}.

A deterministic oblivious sketch may result in estimator degeneration for some data matrix which can be formalized as a theorem.

Theorem 3. *Let $\Pi \in \mathbb{R}^{N\times S}$ be a fixed matrix. There exist Ξ and Y^{f} such that $\tilde{J} = \|Y^{\mathrm{f}}\|_{\mathrm{F}}^2$ and $\hat{J} = 0$.*

Proof. See Appendix . ∎

This theorem indicates that for any data-oblivious deterministic sketch matrix Π, in the worst case, the estimator degenerates into a trivial estimator $\tilde{\Phi} = 0$. In the meantime the dynamics can still be recovered when using the full data.

When the deterministic sketch matrix Π is not data oblivious, it depends on the data matrix Ξ and Y^{f}. Then, from (14) we have $\tilde{\Phi} = \tilde{Y}^{\mathrm{f}}\tilde{\Xi}^{\dagger}$. Substituting this into (15), one may determine the optimal Π by solving the optimization problem

$$\min_{\Pi\in\mathbb{R}^{N\times S}} \|Y^{\mathrm{f}}\Pi(\Xi\Pi)^{\dagger}\Xi - Y^{\mathrm{f}}\|_{\mathrm{F}}^2. \quad (19)$$

Unfortunately, this optimization problem is non-convex and does not have a closed-form solution. Instead, one may apply SVD to the data matrix

$$\begin{bmatrix} \Xi \\ Y^{\mathrm{f}} \end{bmatrix} = U\Lambda V^{\top}, \quad (20)$$

where U and V are orthogonal matrices, and Λ is a diagonal matrix with descending singular values. Then one may choose the first S columns of V as the sketch matrix:

$$\Pi^{\mathrm{svd}} = V_S, \quad \begin{bmatrix} \Xi^{\mathrm{svd}} \\ Y^{\mathrm{f,svd}} \end{bmatrix} = U_S\Lambda_S, \quad (21)$$

where U_S is the first S columns of U, and Λ_S is the first $S\times S$ block of Λ. We call this *SVD sketch*.

When the data does not contain noise, one may choose S such that the S-th largest singular value is above some threshold [8]. However, in the presence of noise, such choice is arbitrary; see already Fig. 1 in Section IV. Moreover, obtaining sketch matrix Π^{svd} takes $\mathcal{O}((d+pT^{\mathrm{f}})N)$ space and each SVD calculation has time complexity $\mathcal{O}((d+pT^{\mathrm{f}})^2N)$ that grow linearly with N.

To avoid the estimator degeneration of the window sketch and the computational issues of the SVD sketch, we propose a new *Gaussian memory sketch*. We rewrite (11) as

$$\tilde{\Xi} = \begin{bmatrix} \xi_1 & \cdots & \xi_N \end{bmatrix} \begin{bmatrix} \pi_1^{\top} \\ \vdots \\ \pi_N^{\top} \end{bmatrix} = \sum_{i=1}^{N} \xi_i\pi_i^{\top}, \quad \tilde{Y}^{\mathrm{f}} = \sum_{i=1}^{N} y_i^{\mathrm{f}}\pi_i^{\top}. \quad (22)$$

If the π_i are mutually independent, we can achieve an incremental update of $\tilde{\Xi}$ and $\tilde{Y}^{\mathrm{f}}$. Namely, when the i-th data ξ_i and y_i^{f} are collected, we update the sketch as

$$\tilde{\Xi} \leftarrow \tilde{\Xi} + \xi_i\pi_i^{\top}, \qquad \tilde{Y}^{\mathrm{f}} \leftarrow \tilde{Y}^{\mathrm{f}} + y_i^{\mathrm{f}}\pi_i^{\top}. \quad (23)$$

Independence can be achieved by choosing π_i to be an i.i.d. Gaussian random vector. This enables us to state the following lemma; see Lemma 5.3.2 in [21].

Lemma 4 (Johnson–Lindenstrauss lemma)**.** *For any $\epsilon > 0$ and $\delta < 1/2$, let*

$$S = \frac{8}{\epsilon^2}\log\frac{2}{\delta}. \quad (24)$$

Construct the random matrix $\Pi \in \mathbb{R}^{N\times S}$ whose entries are i.i.d. Gaussian random variables $\Pi_{ij} \sim \mathcal{N}\left(0, \frac{1}{S}\right)$. Then for any $x \in \mathbb{R}^N$, with probability at least $1-\delta$, we have

$$(1-\epsilon)\|x\|_2^2 \le \left\|x^{\top}\Pi\right\|_2^2 \le (1+\epsilon)\|x\|_2^2. \quad (25)$$

This lemma indicates that, with high probability, Π maps a vector to a lower dimensional space with bounded norm distortion. In addition, we can achieve an upper bound of the in-sample error of the estimator (14) as stated by the following theorem.

Theorem 5. *Given $0 < \epsilon < 1/2$ and $0 < \delta < 1/2$, let the entries of $\Pi \in \mathbb{R}^{N\times S}$ be i.i.d. Gaussian random variables $\Pi_{ij} \sim \mathcal{N}\left(0, \frac{1}{S}\right)$, where S satisfies (24). With probability at least $1-\delta$, for any $\Xi \in \mathbb{R}^{d\times N}$ and $Y^{\mathrm{f}} \in \mathbb{R}^{pT^{\mathrm{f}}\times N}$, we have*

$$\left\|\tilde{\Phi}\Xi - Y^{\mathrm{f}}\right\|_{\mathrm{F}}^2 \le \frac{1+\epsilon}{1-\epsilon}\left\|\hat{\Phi}\Xi - Y^{\mathrm{f}}\right\|_{\mathrm{F}}^2. \quad (26)$$

Proof. By the optimality of $\tilde{\Phi}$ in (14), we have

$$\begin{aligned}\left\|\tilde{\Phi}\tilde{\Xi}-\tilde{Y}^{\mathrm{f}}\right\|_{\mathrm{F}}^{2} &\leq \left\|\hat{\Phi}\tilde{\Xi}-\tilde{Y}^{\mathrm{f}}\right\|_{\mathrm{F}}^{2}=\left\|\left(\hat{\Phi}\Xi-Y^{\mathrm{f}}\right)\Pi\right\|_{\mathrm{F}}^{2} \\ &\leq (1+\epsilon)\left\|\hat{\Phi}\Xi-Y^{\mathrm{f}}\right\|_{\mathrm{F}}^{2}.\end{aligned} \tag{27}$$

On the other hand,

$$\left\|\tilde{\Phi}\tilde{\Xi}-\tilde{Y}^{\mathrm{f}}\right\|_{\mathrm{F}}^{2}=\left\|\left(\tilde{\Phi}\Xi-Y^{\mathrm{f}}\right)\Pi\right\|_{\mathrm{F}}^{2}\geq(1-\epsilon)\left\|\tilde{\Phi}\Xi-Y^{\mathrm{f}}\right\|_{\mathrm{F}}^{2} \tag{28}$$

Combining (27) and (28) we get (26). ∎

This theorem compares the in-sample error of the estimator with Gaussian memory sketch to that of the estimator trained with full data. Notice that the ratio $(1+\epsilon)/(1-\epsilon)$ is independent of the data matrix Ξ and Y^{f}. Moreover, (24) provides a quantitative connection between the memory size S and the prediction accuracy, which only depends on ϵ and δ and is independent of the dimensions of the Hankel matrix d and N. In addition, the logarithmic relationship with δ makes it 'cheap' to achieve a high probability guarantee. Although the worst case cannot be completely avoided, we provide a probabilistic guarantee on the quality of the sketched solution. In addition, such sketching can be implemented incrementally, and we do not need to keep track of the π_i in each iteration. Thus the Gaussian memory sketch achieves constant space and time complexity in each update.

IV. Trajectory Prediction in Traffic

In this section, we demonstrate the effectiveness of the proposed memory sketching for data-driven prediction by applying it to trajectory prediction in vehicular traffic.

In Fig. 1(a), the ego vehicle (brown) is following a chain of vehicles. It can access the speed of vehicle 1 (blue) via a range sensor and the speed of vehicle 5 (purple) through vehicle-to-vehicle (V2V) communication [22]. In [23], we provided a data-driven control framework to predict the speed v_1 based on the speed v_5. Here we apply memory sketching to obtain data-driven predictions with high accuracy while using limited memory space.

The speed v_5 is generated from a Gaussian process with Matérn kernel

$$k_\nu(x,x')=\sigma^2\frac{2^{1-\nu}}{\Gamma(\nu)}\left(\sqrt{2\nu}\frac{\|x-x'\|}{L}\right)^{\nu}K_\nu\left(\sqrt{2\nu}\frac{\|x-x'\|}{L}\right), \tag{29}$$

where K_ν is the modified Bessel function of the second kind. Here we choose $\nu=3/2$ so that the sample paths are differentiable [24]. The length scale is $L=6$ [s] and the magnitude is $\sigma=3$ [m/s]. We simulate the speeds v_4, v_3, v_2, v_1 using the delayed optimal velocity model with additive noise:

$$\begin{aligned}\dot{h}_i(t) &= v_{i+1}(t)-v_i(t),\\ \dot{v}_i(t) &= \alpha\big(\kappa(h_i(t-\tau)-h_{\min})-v(t-\tau)\big),\\ &\quad+\beta\big(v_{i+1}(t-\tau)-v_i(t-\tau)\big)+w(t),\end{aligned} \tag{30}$$

cf. [22]. Here $w(t)$ is a Gaussian white noise, τ is the reaction delay of drivers, $h_i=s_{i+1}-s_i-l$ is the headway, and $l=5$ [m] the vehicle length. Note that (30) is an affine system so linearizing it around the equilibrium $h_i(t)\equiv v^*/\kappa+h_{\min}$, $v_i(t)\equiv v^*$ results in an LTI system. When generating the data we discretize time with step size $\Delta t=0.1$ [s] using Euler's scheme and we choose the parameters $v_{\max}=35$ [m/s], $h_{\min}=5$ [m], $\alpha=0.2$ [1/s], $\beta=0.8$ [1/s], $\kappa=0.6$ [1/s], $\tau=0.8$ [s] and $v^*=30$ [m/s].

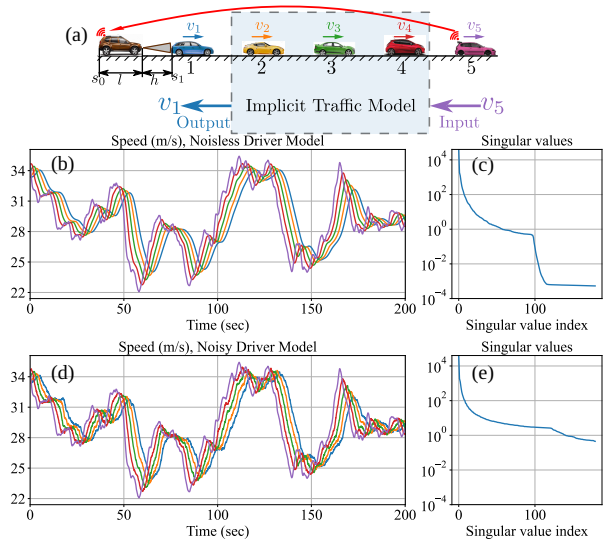

Fig. 1. (a) Implicit traffic model with speed v_5 as input and speed v_1 as output. (b) Speed trajectories generated with (30) without noise. (c) Singular values of the data matrix without noise. (d) Speed trajectories generated with (30) with noise. (e) Singular values of the data matrix with noise.

In Fig. 1(b) and (d), we plot the trajectories generated with a model without noise and with noise of standard deviation 0.5 [$\mathrm{m}^2/\mathrm{s}^2$]. This noise value is so small that it is difficult to distinguish the trajectories between the two panels. Yet, when computing the singular values of the corresponding data matrices they look significantly different; cf. Fig. 1(c) and (e). Without noise, there is a rapid drop around the 100th singular value, while with noise no clear threshold can be found. In the rest of the paper, we use the data with noise.

Speed trajectories of length 1000 [s] are generated for vehicles 1 and 5. We break the trajectories into two phases. In the first, observation phase, we observe and sketch the trajectories without making predictions. In the second, evaluation phase, new observation data is collected in every 2 seconds, the memory is sketched, and a prediction is made. The prediction horizon is $T^{\mathrm{f}}=5/\Delta t=50$. We evaluate the performance of three memory sketching methods (window sketch, SVD sketch, Gaussian sketch), and compare them with the baseline predictor, which uses all the available data for the prediction. At time step k, we record the predicted speed $\hat{v}_1(k+i)$, $i=1,\ldots,T^{\mathrm{f}}$. The normalized square error

$$E_k=\frac{1}{T^{\mathrm{f}}}\sum_{i=1}^{T^{\mathrm{f}}}\|v_1(k+i)-\hat{v}_1(k+i)\|_2^2 \tag{31}$$

is used for performance evaluation: we collect E_k, and compare the mean and standard deviation for different methods.

Figure 2 shows the predicted speed trajectories with different sketching methods for two different memory sizes. The purple solid curve depicts the input v_5, and the blue

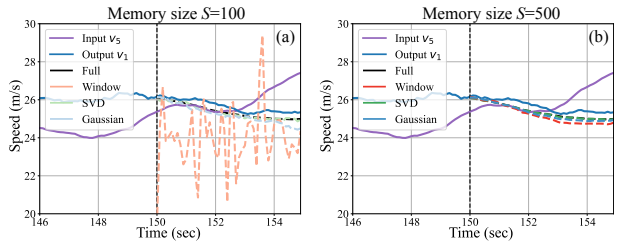

Fig. 2. Comparison of predicted speed trajectories with different sketching methods with memory sizes (a) $S = 100$ and (b) $S = 500$. The purple solid curve shows the input v_5 and the solid blue curve depicts the output v_1. The predictions of v_1 are plotted as dashed curves for the window sketch (orange), SVD sketch (light green), Gaussian memory sketch (light blue), and full-data prediction (black).

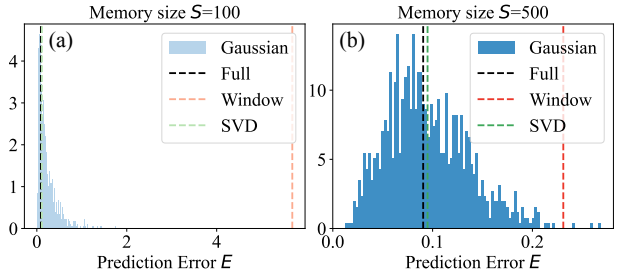

Fig. 3. Histogram of the prediction error (31) for Gaussian memory sketch with different memory sizes (a) $S = 100$ and (b) $S = 500$.

solid curve shows the output v_1. The predictions of v_1 are plotted as dashed curves. The results of the window sketch, SVD sketch, and Gaussian sketch are plotted with orange, green, and blue dashed curves, respectively. The results of the full-data prediction are plotted with black dashed lines.

Due to the randomness of Gaussian sketch, we generate predictions of v_1 using 1000 different Gaussian sketch matrices, and plot the distribution of the prediction error in Fig. 3. With high probability, the Gaussian sketch outperforms the window sketch and on average performs as good as the the SVD sketch and full data. When the memory size is small ($S = 100$ corresponding to 10 seconds of memory), the persistency of excitation is not satisfied for the window sketch due to the limited amount of data, and the dynamics are not captured by the predictions. On the other hand, the full prediction and SVD prediction have access to all the historical observations, thus, the persistency of excitation is satisfied. The Gaussian sketch can capture the dynamics with 10 seconds of memory and incremental update. When the memory size is large ($S = 500$ corresponding to 50 seconds of memory), all the sketching methods capture the dynamics (while the window sketch still performs the worst).

We also investigate the influence of the past horizon T^{p} on the prediction error. Fig. 4(a,c) shows the statistics of prediction error for memory size $S = 100$. The prediction errors for the window sketch are significantly larger than for the other methods. The SVD sketch has a similar performance as the full-data method since they both have access to the full trajectory history. The Gaussian sketch has a slightly larger error but is still in the same order of magnitude. The window sketch is sensitive to the choice of the past horizon,

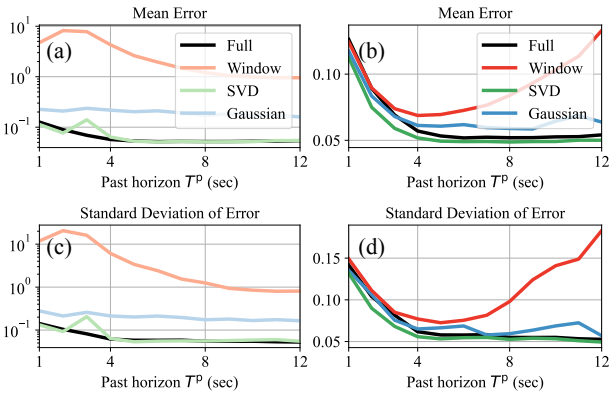

Fig. 4. Statistics of the prediction error (31) as a function of the past horizon for different sketching methods. (a) Mean for memory size $S = 100$, (b) Mean for memory size $S = 500$, (c) Standard deviation for memory size $S = 100$, and (d) Standard deviation for memory size $S = 500$.

while the Gaussian sketch achieves similar errors for all the choices of the past horizon. Initially, the errors of the SVD method and full-data method decrease with the past horizon but they do not decrease further when the past horizon is larger than 4 seconds.

Fig. 4(b,d) shows the statistics of prediction error for memory size $S = 500$. The prediction errors of all the methods are in the same order of magnitude while the window sketch still has the largest error. The SVD sketch and full-data method have similar performance. The Gaussian sketch has a slightly larger error but still in the same order of magnitude. As the prediction horizon increases, the error of the window sketch has a convex shape: it decreases first and then increases with increasing the past horizon. This is because increasing T^{p} increases the complexity of the model while the most recent data does not necessarily contain enough information to capture the dynamics. The error of Gaussian sketch, SVD sketch, and full-data methods decrease at first but when the past horizon reaches 4 seconds they do not further decrease.

We highlight that in Fig. 4, changes occur around 4 seconds of past horizon. This is related to the fact that each driver responds to its predecessor with a delay of about 1 second, yielding a total delay of about 4 seconds for the vehicle chain. This effect is incorporated in classical traffic models [25].

V. Conclusion

In this paper, we established memory sketching for data-driven prediction. We first provided two formulations of the data-driven prediction problem, and then presented three different sketching methods. We showed that the deterministic sketching methods (window sketch and SVD sketch) have a dilemma between data obliviousness and the optimality of the sketch matrix. Then we proposed a randomized sketching method (Gaussian memory sketch) and we showed that it is data oblivious and has optimality guarantee with high probability. It enables incremental updates of memory sketch, and thus, achieves constant space and time complexity. We demonstrated the effectiveness of the proposed memory sketching method on traffic trajectory prediction. We showed

that the sketching method can achieve high prediction accuracy while utilizing limited memory, and thus, it enables online data-driven prediction. We also investigated the influence of past horizons on prediction accuracy. As future research, we are interested in applying memory sketching to time-varying, nonlinear systems, and in how the sketching influences the performance of data-driven predictive control.

APPENDIX

PROOF OF THEOREM 3

Proof. Let the SVD of Π be $\Pi = U\Lambda V^\top$. The notation may be in collision with (20), but in this proof, we focus on the SVD of Π. Define $U^\perp$ such that its column space is the orthogonal complement of the column space of U. When $S \leq N-d$, $\operatorname{rank}(U^\perp) \geq N-d \geq S$, we can choose $\Xi^\top \in \operatorname{range}(U^\perp)$ such that $\Xi U = 0$. Then the sketched least squares problem (14) gives $\tilde{\Phi} = 0$, and the sketched in-sample error is $\tilde{J} = \|Y^{\mathrm{f}}\|_{\mathrm{F}}^2$. We can choose $Y^{\mathrm{f}} \in \operatorname{range}(\Xi^\top)$ to make sure $\hat{J} = 0$.

When $S > N-d$, let $\Xi = [U_1 \;\; U^\perp]^\top$, where U_1 are the last $S-(N-d)$ columns of U. Thus the rows of Ξ are orthogonal. Then we have

$$\begin{aligned}\Xi\Pi &= \begin{bmatrix} U_1 & U^\perp \end{bmatrix}^\top U\Lambda V^\top \\ &= \begin{bmatrix} I_{S-(N-d)} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \Lambda_1 & 0 \\ 0 & \Lambda_2 \end{bmatrix} \begin{bmatrix} V_1^\top \\ V_2^\top \end{bmatrix} \\ &= \begin{bmatrix} \Lambda_1 V_1^\top \\ 0 \end{bmatrix},\end{aligned} \tag{32}$$

where Λ_1 has size $(S-(N-d)) \times (S-(N-d))$, and Λ_2 has size $(N-d)\times(N-d)$. Then we have

$$(\Xi\Pi)^\dagger = \begin{bmatrix} V_1\Lambda_1^{-1} & 0 \end{bmatrix}, \tag{33}$$

and the sketched in-sample error is

$$\begin{aligned}\tilde{J} &= \left\|Y^{\mathrm{f}}\Pi(\Xi\Pi)^\dagger\Xi - Y^{\mathrm{f}}\right\|_{\mathrm{F}}^2 \\ &= \left\|Y^{\mathrm{f}}U\Lambda V^\top \begin{bmatrix} V_1\Lambda_1^{-1} & 0 \end{bmatrix} \begin{bmatrix} U_1 & U^\perp \end{bmatrix}^\top - Y^{\mathrm{f}}\right\|_{\mathrm{F}}^2 \\ &= \left\|Y^{\mathrm{f}}\left(U_1U_1^\top - I\right)\right\|_{\mathrm{F}}^2 = \left\|Y^{\mathrm{f}}U_3U_3^\top\right\|_{\mathrm{F}}^2,\end{aligned} \tag{34}$$

(cf. (15),(19)) where $U_3 = U_1^\perp$ is the orthogonal complement of U_1.

On the other hand, since Ξ is a matrix of size $d \times N$ with orthogonal rows, $d < N$, we have

$$\begin{aligned}\hat{J} &= \left\|Y^{\mathrm{f}}\Xi^\dagger\Xi - Y^{\mathrm{f}}\right\|_{\mathrm{F}}^2 = \left\|Y^{\mathrm{f}}(\Xi^\top\Xi - I)\right\|_{\mathrm{F}}^2 \\ &= \left\|Y^{\mathrm{f}}U_4U_4^\top\right\|_{\mathrm{F}}^2,\end{aligned} \tag{35}$$

where $U_4 = \Xi^{\top\perp}$ is the orthogonal complement of $\Xi^\top$. By definition of Ξ, we have $U_3 = [U_4 \;\; U^\perp]$. Notice that $U_3U_3^\top$ and $U_4U_4^\top$ are both orthogonal projection matrices. Let $Y^{\mathrm{f}} \in \operatorname{range}(U^\perp)$, then $\hat{J} = 0$ while $\tilde{J} = \|Y^{\mathrm{f}}\|_{\mathrm{F}}^2$. ■

Remark. *As N increases, the dimension of $U^\perp$ becomes larger, then the intersection of the range space of data matrix Ξ and $U^\perp$ is non-empty with higher probability, so $\tilde{J}$ will be large with higher probability.*

REFERENCES

[1] J. Wang, Y. Zheng, K. Li, and Q. Xu, "DeeP-LCC: Data-enabled predictive leading cruise control in mixed traffic flow," *IEEE Transactions on Control Systems Technology*, pp. 1–17, 2023.

[2] E. Elokda, J. Coulson, P. N. Beuchat, J. Lygeros, and F. Dörfler, "Data-enabled predictive control for quadcopters," *International Journal of Robust and Nonlinear Control*, vol. 31, no. 18, pp. 8916–8936, 2021.

[3] L. Huang, J. Coulson, J. Lygeros, and F. Dörfler, "Data-enabled predictive control for grid-connected power converters," in *58th IEEE Conference on Decision and Control (CDC)*, 2019, pp. 8130–8135.

[4] I. Markovsky, L. Huang, and F. Dörfler, "Data-driven control based on the behavioral approach: From theory to applications in power systems," *IEEE Control Systems Magazine*, vol. 43, no. 5, pp. 28–68, 2023.

[5] S. Baros, C.-Y. Chang, G. E. Colon-Reyes, and A. Bernstein, "Online data-enabled predictive control," *Automatica*, vol. 138, p. 109926, 2022.

[6] J. Shi and C. N. Jones, "Efficient recursive data-enabled predictive control: an application to consistent predictors," *arXiv preprint arXiv:2309.13755*, 2023.

[7] L. Schmitt, J. Beerwerth, M. Bahr, and D. Abel, "Data-driven predictive control with online adaption: Application to a fuel cell system," *IEEE Transactions on Control Systems Technology*, 2023.

[8] K. Zhang, Y. Zheng, C. Shang, and Z. Li, "Dimension reduction for efficient data-enabled predictive control," *IEEE Control Systems Letters*, vol. 7, pp. 3277–3282, 2023.

[9] D. P. Woodruff, "Sketching as a tool for numerical linear algebra," *Foundations and Trends® in Theoretical Computer Science*, vol. 10, no. 1–2, pp. 1–157, 2014.

[10] G. Cormode and S. Muthukrishnan, "An improved data stream summary: the count-min sketch and its applications," *Journal of Algorithms*, vol. 55, no. 1, pp. 58–75, 2005.

[11] E. Dobriban and S. Liu, "Asymptotics for sketching in least squares regression," *Advances in Neural Information Processing Systems*, 2019.

[12] S. Foucart and H. Rauhut, *A Mathematical Introduction to Compressive Sensing*. Springer, 2013.

[13] Y. Yang, M. Pilanci, and M. J. Wainwright, "Randomized sketches for kernels: Fast and optimal nonparametric regression," *The Annals of Statistics*, vol. 45, no. 3, pp. 991 – 1023, 2017.

[14] M. Ghashami, E. Liberty, J. M. Phillips, and D. P. Woodruff, "Frequent directions: Simple and deterministic matrix sketching," *SIAM Journal on Computing*, vol. 45, no. 5, pp. 1762–1792, 2016.

[15] S. Dasgupta and A. Gupta, "An elementary proof of a theorem of Johnson and Lindenstrauss," *Random Structures & Algorithms*, vol. 22, no. 1, pp. 60–65, 2003.

[16] J. C. Willems and J. W. Polderman, *Introduction to mathematical systems theory: a behavioral approach*. Springer, 1997.

[17] J. C. Willems, P. Rapisarda, I. Markovsky, and B. L. De Moor, "A note on persistency of excitation," *Systems & Control Letters*, vol. 54, no. 4, pp. 325–329, 2005.

[18] I. Markovsky and F. Dörfler, "Behavioral systems theory in data-driven analysis, signal processing, and control," *Annual Reviews in Control*, vol. 52, pp. 42–64, 2021.

[19] H. J. van Waarde, C. De Persis, M. K. Camlibel, and P. Tesi, "Willems' fundamental lemma for state-space systems and its extension to multiple datasets," *IEEE Control Systems Letters*, vol. 4, no. 3, pp. 602–607, 2020.

[20] B. Huang and R. Kadali, *Dynamic modeling, predictive control and performance monitoring: a data-driven subspace approach*. Springer, 2008.

[21] R. Vershynin, *High-dimensional probability: An introduction with applications in data science*. Cambridge University Press, 2018.

[22] M. Shen, C. R. He, T. G. Molnár, A. H. Bell, and G. Orosz, "Energy-efficient connected cruise control with lean penetration of connected vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 4, pp. 4320–4332, 2023.

[23] M. Shen and G. Orosz, "Data-driven predictive connected cruise control," in *IEEE Intelligent Vehicles Symposium (IV)*, 2023, pp. 1–6.

[24] C. K. Williams and C. E. Rasmussen, *Gaussian processes for machine learning*. MIT Press, 2006, vol. 2, no. 3.

[25] T. G. Molnár, D. Upadhyay, M. Hopka, M. Van Nieuwstadt, and G. Orosz, "Delayed Lagrangian continuum models for on-board traffic prediction," *Transportation Research Part C*, vol. 123, p. 102991, 2021.