Statistically and Computationally Efficient Variance Estimator for Kernel Ridge Regression

Meimei Liu
Department of Statistical Science
Duke University
Durham, IN - 27708
Email: meimei.liu@duke.edu

Jean Honorio
Department of Computer Science
Purdue University
West Lafayette, IN - 47907
Email: jhonorio@purdue.edu

Guang Cheng
Department of Statistics
Purdue University
West Lafayette, IN - 47907
Email: chengg@purdue.edu

Abstract—In this paper, we propose a random projection approach to estimate variance in kernel ridge regression. Our approach leads to a consistent estimator of the true variance, while being computationally more efficient. Our variance estimator is optimal for a large family of kernels, including cubic splines and Gaussian kernels. Simulation analysis is conducted to support our theory.

I. INTRODUCTION

As a flexible nonparametric tool, kernel ridge regression (KRR) has gained popularity in many application fields, such as machine learning, and visualization; see e.g., [11]. Besides the estimation of the predictive mean, an exploration of the *predictive variance* is also important for statistical inference. Predictive variances can be used for inference, for example, to build confidence intervals; or to select the most informative data points in active learning. There are two sources of uncertainty in the predictive variance: the noise in the data and the uncertainty in the estimation of the target function. However, calculating the second uncertainty is challenging in KRR on a large data set, since the computational burden increases dramatically with respect to the size of the training set. For example, for n data points, the time and space complexity of kernel ridge regression (KRR) are of $O(n^3)$ and $O(n^2)$ respectively. The above can potentially limit the applicability of KRR to big data scenarios.

An efficient way to break the computational bottleneck is low-rank approximation of kernel matrices. Existing methods include dimension reduction ([16], [3], [2]), Nyström approximation ([12], [1], [10]), and random projections of large kernel matrices ([15]). Indeed, low-rank approximation strategies effectively reduce the size of large matrices such that the reduced matrices can be conveniently stored and processed.

In this paper, we propose a randomly sketched predictive variance to reduce the computational complexity. Theoretically, we show that given a lower bound of the projection dimension, our approach leads to a consistent estimator of the true variance. Furthermore, our variance estimator is optimal for a large family of kernel matrices with polynomially and exponentially decaying eigenvalues. This includes, for instance, cubic splines and Gaussian kernels.

To illustrate the applicability of our theorical contribution, we describe an application of our variance estimator in active learning. In many scenarios, the task of manually labeling (unlabeled) data points is expensive and time-consuming. Therefore it is very important to minimize the number of training examples needed to estimate a particular type of regression function. Suppose we have a set of training examples and labels (responses), and we are permitted to actively choose future unlabeled examples based on the data that we have previously seen. Active learning aims at solving this problem and has been used with various learners such as neural networks [9], [5], mixture models [6], support vector machines [13] and kernel ridge regression. In active learning, the predictive variance can be viewed as an uncertainty score to iteratively select the most informative unlabeled data points. That is, the largest predictive variance corresponds to the highest uncertainty in y for unlabeled points, which indicates that we may need more information regarding those points.

II. PRELIMINARIES

In this section, we introduce kernel ridge regression, its mean prediction and the conditional covariance. Let $\mathbf{X} = (X_1, \cdots, X_n)^\top$ be the training examples, and $\boldsymbol{y} = (y_1, \cdots, y_n)$ be the corresponding training labels, where $X_i \in \mathcal{X}$ with distribution P_X and $y_i \in \mathcal{R}$ for all $i = 1, \ldots, n$. Consider the following nonparametric regression model

$$y_i = f^*(X_i) + \epsilon_i, \quad \text{for } i = 1, \cdots, n \quad (1)$$

where ϵ_i's are independent random variables with mean 0 and variance σ^2. Hereafter, we assume that $\mathcal{H}$ is a reproducing kernel Hilbert space (RKHS) associated with a reproducing kernel function $K(\cdot, \cdot)$ defined from $\mathcal{X} \times \mathcal{X}$ to $\mathbb{R}$. Let $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ denote the inner product of $\mathcal{H}$ associated with $K(\cdot, \cdot)$, then the reproducing kernel property states that

$$\langle f, K(x, \cdot) \rangle_{\mathcal{H}} = f(x), \quad \text{for all } f \in \mathcal{H}.$$

The corresponding norm is defined as $\|f\|_{\mathcal{H}} := \sqrt{\langle f, f \rangle_{\mathcal{H}}}$ for any $f \in \mathcal{H}$.

The classic kernel ridge regression (KRR) estimate is obtained via minimizing a penalized likelihood function:

$$\widehat{f}_n \equiv arg\,min_{f \in \mathcal{H}} \left\{ \frac{1}{n} \sum_{i=1}^n (y_i - f(X_i))^2 + \lambda \|f\|_{\mathcal{H}}^2 \right\} \quad (2)$$

Let K be the n-dimensional kernel matrix with entries $K_{ij} = \frac{1}{n}K(X_i, X_j)$ for $1 \leq i, j \leq n$. By the representer theorem, $\widehat{f}_n$ has the form

$$f(\cdot) = \sum_{i=1}^{n} \omega_i K(\cdot, X_i)$$

for a real vector $\omega = (\omega_1, \ldots, \omega_n)^\top$, equation (2) reduces to solving the following optimization problem:

$$\omega^\dagger = arg\,min_{\omega \in \mathbb{R}^n}\left\{\omega^\top K^2 \omega - \frac{2}{n}\boldsymbol{y}^\top K\omega + \lambda \omega^\top K\omega\right\}. \quad (3)$$

Thus, the KRR estimator is expressed as $\widehat{f}_n(\cdot) = \sum_{i=1}^{n}\omega_i^\dagger K(\cdot, X_i)$, where $\omega^\dagger = \frac{1}{n}(K + \lambda I)^{-1}\boldsymbol{y}$.

For a new testing data point x, let $k(x) = (K(x, X_1), K(x, X_2), \cdots, K(x, X_n))^\top$. It is easy to calculate its mean prediction and variance given $\mathbf{X}$ and $\boldsymbol{y}$ as follows:

$$\widehat{y}(x) = \frac{1}{n}k(x)^\top (K + \lambda I)^{-1}\boldsymbol{y}$$

$$V_1(x) = \text{Var}(\widehat{y}(x)|\mathbf{X}, x) = \frac{\sigma^2}{n^2}k(x)^\top (K + \lambda I)^{-2}k(x).$$

Analyzing the conditional variance for the testing data is very important in active learning, since it can act as a guide to select the efficient information we need. However, the time and space taken for solving $(K + \lambda I)^{-1}$ is of order $O(n^3)$. This cost is expensive especially when the kernel matrix is dense and the sample size is large.

III. RANDOMLY PROJECTED VARIANCE

In this section, we introduce our randomly projected conditional covariance and our main assumptions. Note that by the Binomial Inverse Theorem, we have

$$(K + \lambda I)^{-1} = \frac{1}{\lambda}(I - K(\lambda K + K^2)^{-1}K). \quad (4)$$

To reduce the computational cost, now we propose to replace $(K + \lambda I)^{-1}$ by using a randomly projected version as follows

$$\frac{1}{\lambda}(I - KS^\top(\lambda SKS^\top + SK^2S^\top)^{-1}SK), \quad (5)$$

where $S \in \mathbb{R}^{m \times n}$ is a random matrix where each row is independently distributed and sub-Gaussian. Then the conditional variance with the randomly projected matrix can be written as

$$V_2(x) = \text{Var}(\widehat{y}(x)|\mathbf{X}, x, S)$$
$$= \frac{\sigma^2}{n^2\lambda^2}k(x)^\top(I - KS^\top(\lambda SKS^\top + SK^2S^\top)^{-1}SK)^2 k(x) \quad (6)$$

The definition of V_2 is also our contribution. The variance V_2 is different from the variance that could be derived from the results in [15], which is:

$$V_3(x) = \frac{\sigma^2}{n^2}k(x)^\top KS^\top(\lambda SKS^\top + SK^2S^\top)^{-1}SK^2S^\top$$
$$(\lambda SKS^\top + SK^2S^\top)^{-1}SKk(x)$$

Unfortunately, understanding the concentration of V_3 seems highly nontrivial. However, our new proposed randomly

sketched variance V_2 in eq.(6) has nice concentration properties in Theorem IV.1.

Note that calculating $(\lambda SKS^\top + SK^2S^\top)^{-1}$ only takes the order of $\mathcal{O}(mn^2)$, which enhances the computational efficiency greatly. In Section 4, we will provide a lower bound for m that guarantees stability of the variance after random projection.

Here we introduce some notations to study the dimension of the random matrix. Define the efficiency dimension as

$$s_\lambda = arg\,min\{j : \widehat{\mu}_j \leq \lambda\} - 1, \quad (7)$$

where $\widehat{\mu}_j$ is the j-th highest eigenvalue of the kernel matrix K. More formally, let $K = UDU^\top$, where $U \in \mathbb{R}^{n \times n}$ is an orthonormal matrix, i.e., $UU^\top$ is an $n \times n$ identity matrix, and $D \in \mathbb{R}^{n \times n}$ is a diagonal matrix with diagonal elements $\widehat{\mu}_1 \geq \widehat{\mu}_2 \geq \cdots \geq \widehat{\mu}_n > 0$.

In this paper, we consider random matrices with independent sub-Gaussian rows. For the random matrix S, the ith row $S_i \in \mathbb{R}^n$ is sub-Gaussian if for all $u \in \mathbb{R}^n$, $\langle S_i, u \rangle$ are sub-Gaussian random variables, i.e.,

$$\mathbb{P}\{|\langle S_i, u \rangle| > t\} \leq e \cdot \exp\{-t^2\}.$$

Matrices fulfilling the above condition include all matrices with independent sub-Gaussian entries as a particular instance. The class of sub-Gaussian variates includes for instance Gaussian variables, any bounded random variable (e.g. Bernoulli, multinomial, uniform), any random variable with strongly log-concave density, and any finite mixture of sub-Gaussian variables. In the following of the paper, we scale the random matrix by $\sqrt{m}$ for analyzing convenience.

Next, we state our main assumption and some useful results related to the randomly projected kernel matrix.

Assumption A1. *Let S be a sub-Gaussian random matrix with independent rows. Let $\lambda \to 0$ and $\lambda \gg 1/n$. Set the projection dimension $m \geq ds_\lambda$, where d is an absolute constant. For $K = UDU^\top$, let $U = (U_1, U_2)$ with $U_1 \in \mathbb{R}^{n \times s_\lambda}$, and $U_2 \in \mathbb{R}^{n \times (n-s_\lambda)}$. Let $D = \begin{pmatrix} D_1 & 0 \\ 0 & D_2 \end{pmatrix}$ with $D_1 \in \mathbb{R}^{s_\lambda \times s_\lambda}$ and $D_2 \in \mathbb{R}^{(n-s_\lambda) \times (n-s_\lambda)}$. We assume that S satisfies the following conditions:*

(i) $1/2 \leq \lambda_{\min}(SU_1) \leq \lambda_{\max}(SU_1) \leq 3/2$ with probability greater than $1 - 2\exp\{-cm\}$, where c is an absolute constant independent of n.

(ii) $\|SU_2D_2^{1/2}\|_{op} \leq c'\lambda^{1/2}$ with probability greater than $1 - 2\exp\{-c''m\}$, where c' and c'' are constants independent of n.

In Assumption A1, the kernel matrix is partitioned into a summation of the form $K = U_1 D_1 U_1^\top + U_2 D_2 U_2^\top$ where U_1 contains the first s_λ columns of the orthonormal matrix U, which correspond to the first leading eigenvalues of the kernel matrix; and U_2 contains the rest of the $n - s_\lambda$ columns of the U, which correspond to the smallest $n - s_\lambda$ eigenvalues. In most cases, the smallest $n - s_\lambda$ eigenvalues are neglectable due to a fast decaying rate of the eigenvalues. Assumption A1 (i) ensures that the randomly projected eigenvectors corresponding

to the leading eigenvalues still preserve the distance between each other approximately; Assumption A1 (ii) ensures that the operator norm of the lowest "neglectable" part would not change too much after the random projection. Random matrices satisfying Assumption A1 include sub-Gaussian random matrices (see detailed proof in [8]), as well as matrices constructed by randomly sub-sampling and rescaling the rows of a fixed orthonormal matrix. We refer the interested reader to [15], [14] for more details.

Our work differs from [15] in several fundamental ways. [15] focuses on the (mean) prediction error on a training set, and it is unclear how this relates to a prediction error on a testing set. In contrast to [15], we target the variance of the prediction error, and focus on prediction on a test set. Additionally, note that we define s_λ by the tuning parameter λ as in eq. (7), which is different from [15].

IV. MAIN RESULTS

Recall that for a new testing data x, the conditional variance given training data $\mathbf{X}$ and $\boldsymbol{y}$ is $V_1(x)$. In this section, we will show that for the new data x, our new proposed randomly sketched conditional variance $V_2(x)$ can provide a stable approximation for the original conditional covariance $V_1(x)$.

Theorem IV.1. *Under Assumption A1, suppose $\lambda \to 0$ as $n \to \infty$, $\lambda \gg n^{-1}$, and the projection dimension $m \geq ds_\lambda$. Then with probability at least $1 - 2\exp(-cm)$, with respect to the random choice of S, we have*

$$\sup_{x \in \mathcal{X}} |V_1(x) - V_2(x)| \leq \frac{c'\sigma^2}{n\lambda},$$

where c and c' are absolute constants independent of n.

As shown in Theorem IV.1, the convergence rate involves λ directly. Normally, we choose λ as the optimal one to achieve minimax optimal estimation. Next, we provide some examples to show how to choose the lower bound of the projection dimension for the random matrix S and the corresponding optimal λ.

Proof. Let $k(x) = (g(X_1), \cdots, g(X_n))^\top$ with $g(X_i) = K(x, X_i) = \langle K(x, \cdot), K(\cdot, X_i)\rangle$, then we have that $g(\cdot) = K(x, \cdot) \in \mathcal{H}$. We denote $g^* = k(x) = (g(X_1), \cdots, g(X_n))^\top$, and let $\sigma = 1$. Then $V_1(x)$ can be written as

$$V_1(x) = \frac{\sigma^2}{n^2\lambda^2}\|(I - K(\lambda K + K^2)^{-1}K)g^*\|_2^2,$$

by eq.(4), where $\|\cdot\|_2$ is the Euclidean norm. Furthermore

$$V_2(x) = \frac{\sigma^2}{n^2\lambda^2}\|(I - KS^\top(\lambda SKS^\top + SK^2S^\top)^{-1}SK)g^*\|_2^2.$$

and therefore:

$$\begin{aligned}
&\sup_{x \in \mathcal{X}} |V_1(x) - V_2(x)|\\
&= \sup_{x \in \mathcal{X}} \frac{\sigma^2}{n^2\lambda^2}[(I - K(\lambda K + K^2)^{-1}K)g^*\\
&\quad + (I - KS^\top(\lambda SKS^\top + SK^2S^\top)^{-1}SK)g^*]^\top\\
&\quad \cdot [(I - K(\lambda K + K^2)^{-1}K)g^*\\
&\quad - (I - KS^\top(\lambda SKS^\top + SK^2S^\top)^{-1}SK)g^*]\\
&\leq \frac{\sigma^2}{n\lambda^2}(T_1 + T_2)^2
\end{aligned} \tag{8}$$

where in the last step, we used Cauchy Schwarz inequality and the triangle inequality. In the above, T_1 and T_2 are defined as follows:

$$T_1 = \frac{1}{\sqrt{n}}\|g^* - K(\lambda K + K^2)^{-1}Kg^*\|_2$$

$$T_2 = \frac{1}{\sqrt{n}}\|g^* - KS^\top(\lambda SKS^\top + SK^2S^\top)^{-1}SKg^*\|_2$$

Next, we prove that

$$T_1^2 \lesssim \lambda, \qquad T_2^2 \lesssim \lambda. \tag{9}$$

Before the proof of eq.(9), we first consider the optimization problem

$$\widehat{\alpha} = arg\,min_{\alpha \in \mathbb{R}^m} \frac{1}{n}\|g^* - nKS^\top\alpha\|_2^2 + n\lambda\|K^{1/2}S^\top\alpha\|_2^2, \tag{10}$$

which has the solution $\widehat{\alpha} = \frac{1}{n}(\lambda SKS^\top + SK^2S^\top)^{-1}SKg^*$. In this case $T_2^2 = \frac{1}{n}\|g^* - nKS^\top\widehat{\alpha}\|_2^2$.

Therefore, to prove $T_2^2 \leq \lambda$, we only need to find a vector $\widetilde{\alpha}$, such that $\frac{1}{n}\|g^* - nKS^\top\widetilde{\alpha}\|_2^2 + n\lambda\|K^{1/2}S^\top\widetilde{\alpha}\|_2^2 \leq c_1\lambda$. This will imply

$$\begin{aligned}
&\frac{1}{n}\|g^* - nKS^\top\widehat{\alpha}\|_2^2 + n\lambda\|K^{1/2}S^\top\widehat{\alpha}\|_2^2\\
&\leq \frac{1}{n}\|g^* - nKS^\top\widetilde{\alpha}\|_2^2 + n\lambda\|K^{1/2}S^\top\widetilde{\alpha}\|_2^2 \leq c_1\lambda.
\end{aligned} \tag{11}$$

By definition of s_λ, when $1 \leq j \leq s_\lambda$ then $\widehat{\mu}_j \geq \lambda$ and when $s_\lambda < j \leq n$ then $\widehat{\mu}_j \leq \lambda$. Let $g^* = (g_1^*, g_2^*)$, where $g_1^* \in \mathbb{R}^{s_\lambda}$, and $g_2^* \in \mathbb{R}^{n-s_\lambda}$. Let $z = \frac{1}{\sqrt{n}}U^\top g^* = (z_1, z_2)$ correspondingly. Also, divide D into D_1, D_2, where D_1, D_2 are $s_\lambda \times s_\lambda$ and $(n-s_\lambda) \times (n-s_\lambda)$ dimension diagonal matrix, respectively. Let $\widetilde{S} = (\widetilde{S}_1, \widetilde{S}_2)$, with $\widetilde{S}_1 \in \mathbb{R}^{s \times s_\lambda}$ as the left block and $\widetilde{S}_2 \in \mathbb{R}^{s \times (n-s_\lambda)}$ as the right block. We construct a vector $\widetilde{\alpha}$ by setting $\widetilde{\alpha} = \frac{1}{\sqrt{n}}\widetilde{S}_1(\widetilde{S}_1^\top\widetilde{S}_1)^{-1}D_1^{-1}z_1 \in \mathbb{R}^s$. By plugging $\widetilde{\alpha}$ into eq.(10), we have that

$$\begin{aligned}
&\frac{1}{n}\|g^* - nUD\widetilde{S}^\top\widetilde{\alpha}\|_2^2\\
&= \|z_1 - \sqrt{n}D_1\widetilde{S}_1^\top\widetilde{\alpha}\|_2^2 + \|z_2 - D_2\widetilde{S}_2^\top\widetilde{S}_1(\widetilde{S}_1^\top\widetilde{S}_1)^{-1}D_1^{-1}z_1\|_2^2\\
&= G_1^2 + G_2^2.
\end{aligned}$$

Clearly, in our construction $G_1^2 = 0$, and thus we focus on analyzing G_2. For any $g(\cdot) \in \mathcal{H}$, there exists a vector $\beta \in \mathbb{R}^n$, such that $g(\cdot) = \sum_{i=1}^n K(\cdot, X_i)\beta_i + \xi(\cdot)$, where $\xi(\cdot) \in \mathcal{H}$, and such that ξ is orthogonal to the span of $\{K(\cdot, X_i), i =$

$1, \cdots, n\}$. Therefore, $\xi(X_j) = \langle \xi, K(\cdot, X_j) \rangle = 0$, and $g(X_j) = \sum_{i=1}^{n} K(X_i, X_j)\beta_i$. Thus $g^* = nK\beta$, where K is the empirical kernel matrix. Assume that $\|g\|_{\mathcal{H}} \leq 1$, then

$$n\beta^\top K\beta \leq 1 \quad \Rightarrow n\beta^\top KK^{-1}K\beta^\top \leq 1$$
$$\Rightarrow \frac{1}{n}g^* K^{-1}g^* \leq 1 \quad \Rightarrow \frac{1}{n}g^* UD^{-1}U^\top g^* \leq 1$$

Then, we have the ellipse constraint that $\|D^{-1/2}z\|_2 \leq 1$, where $z = \frac{1}{\sqrt{n}}U^\top g^*$.

Since we have $\|D_1^{-1/2}z_1\|_2 \leq 1$, $\|D_2^{-1/2}z_2\|_2 \leq 1$, which implies $g^{*T}U_2U_2^\top g^* \leq n\lambda$, we have that

$$G_2 \leq \|z_2\|_2 + \|\sqrt{D_2}\|_{\text{op}}\|\sqrt{D_2}\widetilde{S}_2^\top\|_{\text{op}}\|\widetilde{S}_1\|_{\text{op}}\|(\widetilde{S}_1^\top \widetilde{S}_1)^{-1}\|_{\text{op}}$$
$$\cdot \|D_1^{-1/2}\|_{\text{op}}\|D_1^{-1/2}z_1\|_{\text{op}} \leq c\sqrt{\lambda}$$

Therefore, we have $\|z - \sqrt{n}D\widetilde{S}^\top\widetilde{\alpha}\|_2^2 \leq c'\lambda$. For the penalty term,

$$n\widetilde{\alpha}^\top SKS^\top\widetilde{\alpha} \leq z_1^\top D_1^{-1}z_1 + \|z_1^\top D_1^{-\frac{1}{2}}\|_2\|D_1^{-\frac{1}{2}}\|_{\text{op}}$$
$$\cdot \|\widetilde{S}_2\sqrt{D_2}\|\|\sqrt{D_2}\widetilde{S}^\top\|_{\text{op}}\|D_1^{-\frac{1}{2}}\|_{\text{op}}\|D_1^{-\frac{1}{2}}z_1\|_{\text{op}} \leq c'',$$

where c'' is a constant. Finally, by eq.(11), we can claim that

$$\frac{1}{n}\|KS^\top(\lambda SKS^\top + SK^2S^\top)^{-1}SKg^* - g^*\|_2^2 \leq c_1\lambda,$$

where c_1 is some constant.

Similarly, to prove $T_1^2 \lesssim \lambda$, we can treat S as an identity matrix. Consider the following optimization problem

$$\widehat{w} = arg\,min_{w \in \mathbb{R}^n} \frac{1}{n}\|g^* - nKw\|_2^2 + n\lambda\|K^{1/2}w\|_2^2,$$

which has the solution $\widehat{w} = \frac{1}{n}(\lambda K + K^2)^{-1}Kg^*$. In this case $T_1^2 = \frac{1}{n}\|g^* - nK\widehat{w}\|_2^2$. Therefore, we only need to find a $\widetilde{w}$, such that $\frac{1}{n}\|g^* - nK\widetilde{w}\|_2^2 + n\lambda\|K^{1/2}\widetilde{w}\|_2^2 \leq c_2\lambda$. This will imply

$$\frac{1}{n}\|g^* - nK\widehat{w}\|_2^2 + n\lambda\|K^{1/2}\widehat{w}\|_2^2 \tag{12}$$
$$\leq \frac{1}{n}\|g^* - nK\widetilde{w}\|_2^2 + n\lambda\|K^{1/2}\widetilde{w}\|_2^2 \leq c_2\lambda.$$

Here we construct $\widetilde{w} = \frac{1}{\sqrt{n}}U_1D_1^{-1}z_1$,

$$\frac{1}{n}\|g^* - UDU^\top\widetilde{w}\|_2^2$$
$$= \|z_1 - D_1U_1^\top\widetilde{w}\|_2^2 + \|z_2 - D_2U_2^\top U_1(U_1^\top U_1)^{-1}D_1^{-1}z_1\|_2^2$$
$$= \|z_2\|_2^2 \leq c_2\lambda.$$

For the penalty term, $n\widetilde{w}^\top K\widetilde{w} = z_1^\top D_1^{-1}z_1 \leq 1$. Therefore, combining with eq.(12), we have

$$\frac{1}{n}\|K(\lambda K + K^2)^{-1}Kg^* - g^*\|_2^2 \leq c_2\lambda.$$

Finally, by eq.(4.1), we have

$$\sup_{x \in \mathcal{X}} |V_1(x) - V_2(x)| \leq \frac{\sigma^2}{n\lambda^2}(T_1 + T_2)^2 \lesssim \frac{\sigma^2}{n\lambda}$$

$\square$

Example 1: Consider kernels with polynomially decaying eigenvalues $\mu_k \asymp k^{-2\alpha}$ for $\alpha \geq 1$. Such kernels include the α−order periodic Sobolev space, for $\alpha = 2$, which corresponds to the cubic spline. Since the optimal rate of λ to achieve the minimax estimation error is of order $n^{-\frac{2\alpha}{2\alpha+1}}$, we get the corresponding optimal lower bound for the projection dimension $m \gtrsim s_\lambda \asymp n^{\frac{1}{2\alpha+1}}$. Furthermore, the difference between original conditional variance and randomly sketched conditional variance $|V_1(x) - V_2(x)|$ can be bounded by the order of $O(n^{-\frac{1}{2\alpha+1}})$.

Example 2: Consider kernels with exponentially decaying eigenvalues $\mu_k \asymp e^{-\alpha k^p}$ for $p > 0$, which include the Gaussian kernel with $p = 2$. Since the optimal rate of λ to achieve the minimax estimation rate is of order $(\log n)^{1/p}/n$, we get the corresponding lower bound $m \geq s_\lambda \asymp (\log(n))^{1/p}$, and $|V_1(x) - V_2(x)| \lesssim (\log n)^{-1/p}$.

V. EXPERIMENTS

In this section, we verify the validity of our theoretical contribution (Theorem IV.1) through synthetic experiments.

Data were generated based on eq.(1) with the predictor X following a uniform distribution on $[0, 1]$, $f^*(x) = -1 + 2x^2$, and $\epsilon_i \sim N(0, \sigma^2)$. We used Gaussian random projection matrices.

For the polynomial kernel, Figure 1 (a) shows the gap $\sup_{x \in [0,1]} |V_1(x) - V_2(x)|$ with the training sample size n ranging from 50 to 1000, while fixing $\sigma = 1$, and the projection dimension $m = \lceil 1.5n^{1/(2\alpha+1)} \rceil$ with $\alpha = 2$. Note that with the increase of the sample size n, the gap $|V_1 - V_2|$ decreases with the rate $O(1/n)$ as predicted by Theorem IV.1. For Figure 1 (c), we fix the sample size as $n = 1000$, and $\sigma = 1$, while varying the projection dimension $m = \lceil 1.2n^{c/(2\alpha+1)} \rceil$ with c ranging from 0.4 to 1.9. Note that an increase of m leads to a smaller gap $\sup_x |V_1(x) - V_2(x)|$, but this improvement is no longer obvious when $m \geq 1.2n^{1/(2\alpha+1)}$ (or equivalently when c = 1), which is the optimal projection dimension demonstrated in Example 1. In Figure 1 (e), we vary σ from 0.5 to 5, while fixing sample size $n = 1000$ and projection dimension as $\lceil 1.5n^{1/(2\alpha+1)} \rceil$ with $\alpha = 2$; Note that $\sup_x |V_1(x) - V_2(x)|$ increases almost linearly with respect to σ, which is consistent with our theory.

For the Gaussian kernel, Figure 1 (b) shows gap $\sup_{x \in [0,1]} |V_1(x) - V_2(x)|$ with the training sample size n ranging from 50 to 1000, while fixing $\sigma = 1$ and the projection dimension $m = \lceil 2\sqrt{\log(n)} \rceil$. Note that the gap $|V_1 - V_2|$ decreases with the rate $O(1/n)$ as predicted by Theorem IV.1. For Figure 1 (d), we fix the sample size as $n = 1000$, and $\sigma = 1$, while varying the projection dimension $m = \lceil 1.2(\log(n))^{c/2} \rceil$ with c ranging from 0.3 to 1.8. Note that an increase of m leads to a smaller gap $\sup_x |V_1(x) - V_2(x)|$, but this improvement is no longer obvious when $m \geq 1.2(\log(n))^{1/2}$ (or equivalently when c = 1), which is the optimal projection dimension demonstrated in Example 2. In Figure 1 (f), we fix $n = 1000$ and $m = \lceil 2\sqrt{\log(n)} \rceil$, but vary σ from 0.5 to 5. As in the previous experiment, note that $\sup_x |V_1(x) - V_2(x)|$ increases

almost linearly with respect to σ, which is consistent with our theory.

In Appendix VI-B, we show additional synthetic experiments verifying our theoretical contribution. We further illustrate the use of our projected variance estimator in active learning, in synthetic data as well as two real-world datasets. In this illustrative application, by using the randomly sketched predictive variance, the computational complexity is reduced from $O(n^3)$ to $O(mn)$, where m is the projection dimension. Given our theoretical finding, our variance estimator does not sacrifice statistical accuracy.
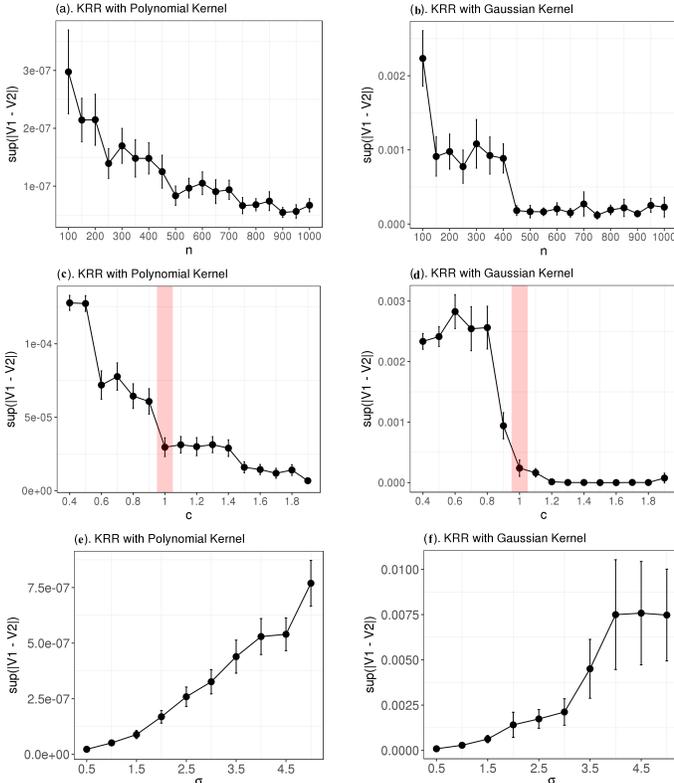
Fig. 1. (a), (c), (e):$\sup_x |V_1(x) - V_2(x)|$ for the polynomial kernel; (b), (d), (f):$\sup_x |V_1(x) - V_2(x)|$ for the Gaussian kernel. Error bars at 95% confidence level for 200 repetitions of the experiments.

VI. CONCLUDING REMARKS

There are several ways of extending this research. While we focused on kernel ridge regression, it would be interesting to propose a statistically and computationally efficient variance estimator for Gaussian processes as well. Additionally, currently in Assumption A1, we only considered sub-Gaussian random matrices, for theoretical convenience. However, the property in Assumption A1 might also hold for subsampled Fourier and Hadamard random matrices, but with a different relationship between m and s_λ. For sub-Gaussian random matrices, we only need $m > s_\lambda$. But for Hadamard random matrices, $m > s_\lambda \log n$ is needed for estimation of the (mean) prediction error as in [15]. This might also likely happen in our predictive variance. But note that our definition of s_λ is different from

[15]. The analysis of different random matrices is appealing for future work. However, our general results on sub-Gaussian matrices should be seen as a necessary first step towards this endeavor.

REFERENCES

[1] Ahmed Alaoui and Michael W Mahoney. Fast randomized kernel ridge regression with statistical guarantees. pages 775–783, 2015.

[2] Maria-Florina Balcan, Avrim Blum, and Santosh Vempala. Kernels as features: On kernels, margins, and low-dimensional mappings. *Machine Learning*, 65(1):79–94, 2006.

[3] Mikio L Braun, Joachim M Buhmann, and Klaus-Robert MÃžller. On relevant dimensions in kernel feature spaces. *Journal of Machine Learning Research*, 9(Aug):1875–1908, 2008.

[4] Klaus Brinker. Incorporating diversity in active learning with support vector machines. In *ICML*, volume 3, pages 59–66, 2003.

[5] David A Cohn. Neural network exploration using optimal experiment design. *Advances in neural information processing systems*, pages 679–679, 1994.

[6] David A Cohn, Zoubin Ghahramani, and Michael I Jordan. Active learning with statistical models. *Journal of artificial intelligence research*, 4(1):129–145, 1996.

[7] James Hensman, Nicolo Fusi, and Neil D Lawrence. Gaussian processes for big data. *UAI*, 2013.

[8] Meimei Liu, Zuofeng Shang, and Guang Cheng. Nonparametric testing under random projection. *arXiv preprint arXiv:1802.06308*, 2018.

[9] David JC MacKay. Information-based objective functions for active data selection. *Neural computation*, 4(4):590–604, 1992.

[10] Cameron Musco and Christopher Musco. Recursive sampling for the nyström method. *arXiv preprint arXiv:1605.07583*, 2016.

[11] Carlotta Orsenigo and Carlo Vercellis. Kernel ridge regression for out-of-sample mapping in supervised manifold learning. *Expert Systems with Applications*, 39(9):7757–7762, 2012.

[12] Alessandro Rudi, Raffaello Camoriano, and Lorenzo Rosasco. Less is more: Nyström computational regularization. pages 1657–1665, 2015.

[13] Simon Tong and Daphne Koller. Support vector machine active learning with applications to text classification. *Journal of machine learning research*, 2(Nov):45–66, 2001.

[14] Roman Vershynin. Introduction to the non-asymptotic analysis of random matrices. *arXiv preprint arXiv:1011.3027*, 2010.

[15] Yun Yang, Mert Pilanci, and Martin J Wainwright. Randomized sketches for kernels: Fast and optimal non-parametric regression. *arXiv preprint arXiv:1501.06195*, 2015.

[16] Tong Zhang. Learning bounds for kernel regression using effective data dimensionality. *Neural Computation*, 17(9):2077–2098, 2005.

APPENDIX

A. Illustrative Application: Randomly Sketched Active Learning Algorithm

Active learning has been successfully applied to classification as well as regression problems [4]. Most active learning algorithms need to iteratively compute a score for each unlabeled samples. Specifically, the kernel ridge regression approach needs to evaluate the prediction variance for the unlabeled samples. As the size of the training data increases, the cost of computation increases cubically. An additional aspect that increases the computational cost is the use of cross validation to select the tuning parameters at each iteration, followed by computing the score for each unlabeled subject. Our randomly sketched active learning is aimed to reduce the computational cost for both model fitting and score calculation.

The main computational cost for randomly sketched KRR lies in computing the matrix multiplication of the sketch matrix and the kernel matrix. Suppose the current training set has n_0 data points and the projection dimension of the

random matrix is m. The computational complexity of the matrix multiplication is of the order of $O(mn_0^2)$. In the next iteration, n_s data points are added to the training set. Instead of calculating the matrix multiplication for all $n_0 + n_s$ data points, we only need to calculate the entries corresponding to the updated data points. We partition the new kernel matrix as

$$K = \begin{bmatrix} K_1 & K_{12} \\ K_{21} & K_2 \end{bmatrix}$$

where $K_1 \in \mathbb{R}^{n_0 \times n_0}$ is the kernel matrix of the current training set, while $K_{12} \in \mathbb{R}^{n_0 \times n_s}$, $K_{21} \in \mathbb{R}^{n_s \times n_0}$ and $K_2 \in \mathbb{R}^{n_s \times n_s}$. Correspondingly, we partition the new random projection matrix as,

$$S = \begin{bmatrix} S_1 & S_{12} \\ S_{21} & S_2 \end{bmatrix}$$

where $S_1 \in \mathbb{R}^{m_1 \times n_0}$ is the sketch matrix from the current step, while $S_{12} \in \mathbb{R}^{m_1 \times n_s}$, $S_{21} \in \mathbb{R}^{(m_2 - m_1) \times n_0}$, $S_2 \in \mathbb{R}^{(m_2 - m_1) \times n_s}$, and m_1 and m_2 are the projections dimension for the current and new sketch matrices correspondingly. Then the matrix multiplication can be written as,

$$
\begin{aligned}
SK &= \begin{bmatrix} K_1 & K_{12} \\ K_{21} & K_2 \end{bmatrix} \begin{bmatrix} S_1 & S_{12} \\ S_{21} & S_2 \end{bmatrix} \\
&= \begin{bmatrix} S_1 K_1 + S_{12} K_{21} & S_1 K_{12} + S_{12} K_{22} \\ S_{21} K_1 + S_2 K_{21} & S_{21} K_{12} + S_2 K_2 \end{bmatrix}
\end{aligned}
\tag{13}
$$

Since $S_1 K_1$ has already been calculated in the previous step, we only need to calculate the remaining terms. The computational complexity is thus reduced from $O(m(n_0 + n_s)^2)$ to $O(mn_0 n_s)$. The size n_0 increases at each iteration, but the step size n_s is fixed. Thus, $O(mn_0 n_s)$ is at most $O(mn_0)$. The reduction of computational complexity is significant for large training sets.

Algorithm 1 Active learning algorithm

Input:
Initial training data set S
Unlabeled data U
repeat
    Step 1: Calculate the projected kernel matrix using eq.(13).

    Step 2: Apply the randomly sketched kernel ridge regression to the training data S.
    Step 3: Calculate the randomly sketched prediction variance $V_2(x)$ for the samples in U as in eq.(3.4).
    Step 4: Sample n_s points based on the weight $V_2(x)$ and obtain the labels associated to them.
    Step 5: Add the sampled points to the training data set S and remove them from unlabeled set U.
until The predefined convergence condition is satisfied
Output:
Final training set S.

The difference between Algorithm 1 and the classical active learning algorithm is that we use $V_2(x)$ as weights to randomly sample data points from the unlabeled training data instead of deterministically selecting the data points with largest scores.

If there is a small cluster of data with large scores, the deterministic method tends to add all of them into the training set initially. Suppose these data points are clustered together and outside of the majority of data points. Once they are all selected in the first few iterations, they may become the majority in the training set, and since the total size of labeled data is very small in early iterations, this will add an extra bias in the prediction. Thus we use the weighted random sampling strategy to ensure a substantial probability to select data points with large score while avoiding to select too many of them at once.

B. Experiments

In this section, we evaluate the performance of our proposed random projection approach. We run experiments on synthetic data as well as on real-world data sets.

1) Confirming our theoretical contribution: Through synthetic experiments, we first verify the validity of our theoretical contribution (Theorem IV.1). Here 500 training samples were generated based on eq.(1) with $X \sim 1/2N(0.5, 0.5) + 1/2N(5, 5)$. We use the Gaussian kernel function $K(x, x') = \exp{-\frac{(x - x')^2}{2\sigma^2}}$, where $\sigma = 1$. Next, we generated 50 testing samples following the same distribution. Here we generated a random projection matrix with Gaussian distributed entries, and the projection dimension is chosen as $m = c\sqrt{\log(n)}$, with $c = 8, 10, 12$ respectively ($m = 20, 25, 30$ approximately). We observe that, with the increase of m, the randomly projected variance performs similar to the original conditional variance, which confirmed the validity of our approach. Also, as for the computational time, the time for calculating the original variance for a new sample x takes $4.654s$, but our proposed new randomly projected variance only takes $0.251s$, showing the practical advantage of our method.
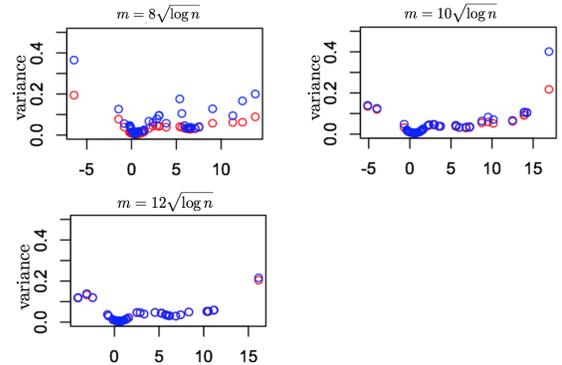
Fig. 2. Red dots represent the original variance for the new sample x, blue dots represent the randomly projected variance.

2) Illustrative application on active learning: synthetic experiments: Next, we illustrate the use of our variance estimator in active learning with synthetic data. (Appendix VI-A provides details of a simple algorithm that uses of our variance estimator and attains $O(mn)$ time.) For comparison, we simulated 5000 data points as the training set and 1000 data points as the testing set. The initial training set was selected

by randomly sampling 100 data points from the training set. In the simulation settings, we use the Gaussian kernel,

$$\mathcal{K}_{gau}(u,v) = e^{-\frac{1}{2h^2}(u-v)^2}$$

with bandwidth $h = 0.25$. We report the mean squared error(MSE) at each iteration.

Simulation Setting 1. We simulate the predictor X from a uniform distribution on $[0,1]$ and $f^*(x) = -1 + 2x^2$. The response y_i was generated as $y_i = f^*(x_i) + \epsilon_i$ $(i = 1, \ldots, n)$, where ϵ_i are i.i.d. standard Gaussian noise.

Gaussian random projection matrix is used in this setting, and we choose the sketch dimension $m = \lceil \log(n) \rceil$. As shown in Figure 2, the randomly sketched active learning algorithm has the smallest mean squared error after 30 iteration. Also, the mean squared error of randomly sketched active learning algorithm converges as fast as the active learning with the original KRR and random sampling with original KRR.

Simulation Setting 2. We simulate the predictor X from the following distribution

$$x_i = \begin{cases} \text{Unif}[0, 1/2] & \text{if } i = 1, \ldots, k \\ 1 + z_i & \text{if } i = k+1, \ldots, n \end{cases}$$

where $z_i \sim N(0, 1/n)$ and $k = \lceil \sqrt{n} \rceil$. For this experiment, we make $f^*(x) = -1 + 2x^2$. The response y_i was generated as $y_i = f^*(x_i) + \epsilon_i$ $(i = 1, \ldots, n)$, where ϵ_i are i.i.d. standard Gaussian noise.

Simulation setting 1   Simulation setting 2
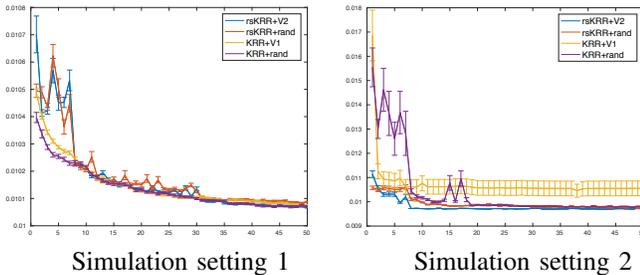
Fig. 3. We compare active learning strategies with random sampling strategies under the original KRR, and the randomly sketched KRR respectively. Y-axes is the mean squared error. At each iteration, we add 30 data points and show the predicted MSE of the four strategies in different colors: "rsKRR+V2" denotes our randomly sketched active learning algorithm, "KRR+V1" denotes active learning with original KRR, "KRR+rand" denotes uniform random sampling with original KRR and "rsKRR+rand" denotes uniform random sampling with randomly sketched KRR. (Error bars at 95% confidence level for 30 repetitions of the experiments.)

Same as the Setting 1, we also use a Gaussian random matrix with sketch dimension $m = \lceil \log(n) \rceil$. As shown in Figure 3, the original active learning method shows faster convergence rate and achieves lower MSE after 50 iterations compared to the random sampling algorithm. For this unevenly distributed data, it is unlikely to select the data outside the majority for the random sampling strategy. However, the minority data with large prediction variance tends to be selected by the active learning algorithm. Thus the randomly sketched active learning algorithm is comparable with active learning with the original KRR after 30 iterations and converges to a similar MSE.

3) Illustrative application on active learning: real-word experiments: Next, we illustrate the use of our variance estimator in active learning with real-world data. (Appendix VI-A provides details of a simple algorithm that uses of our variance estimator and attains $O(mn)$ time.)

Flight Delay Data. Here, we evaluate our randomly sketched active learning algorithm on the US flight dataset [7] that contains up to 2 million points. We use a subset of the data with flight arrival and departure times for commercial flights in 2008. The flight delay was used as our response variable and we included 8 of the many variables from this dataset: the age of the aircraft, distance that needs to be covered, airtime, departure time, arrival time, day of the week, day of the month and month.

Flight delay data   World weather data
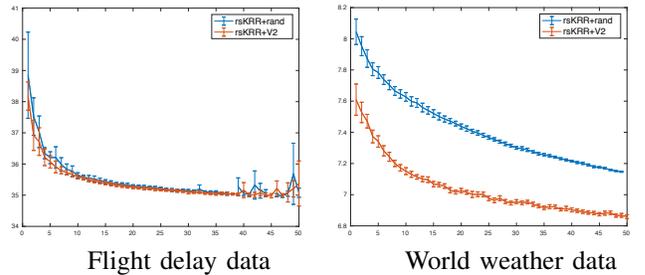
Fig. 4. We compare active learning strategies with random sampling strategies using randomly sketched KRR. Y-axes is the root mean squared error. At each iteration, we add 1000 data points and show the predicted rMSE of two strategies in different colors: "rsKRR+V2" denotes our randomly sketched active learning algorithm, "rsKRR+rand" denotes uniform random sampling with randomly sketched KRR. (Error bars at 95% confidence level for 30 repetitions of the experiments.)

We randomly selected $60,000$ data points, using $50,000$ as the training set and $10,000$ as the testing set. We first randomly selected 1000 data points as labeled data. Then we sequentially added 1000 data points from the unlabeled training data at each iteration. We use the Gaussian random matrix with projection dimension $m = \lceil \log(n) \rceil$. Here we only use the randomly sketched KRR since the computational cost and required RAM of the original KRR is too large. To compare the performance of active learning and uniform sampling, we calculate the RMSE(root mean squared error) 30 times using the prediction on the testing set. In Figure 4, the active learning algorithm achieves the RMSE of the full data faster than the uniform random sampling method.

World Weather Data. In what follows, we examined our method on another real world dataset. The world weather dataset contains monthly measurements of temperature, precipitation, vapor, cloud cover, wet days and frost days from Jan 1990 to Dec 2002 on a 5×5 degree grid that covers the entire world. In our experiments, the response variable is temperature. We use the Gaussian random matrix with projection dimension $m = \lceil \log(n) \rceil$. We use $10,000$ samples for training and $10,000$ samples for testing. We start with an initial set of 200 labeled points, and add 200 points at each iteration. As we can observe in Figure 5, our method compares favorably.