Sketch Kernel Ridge Regression Using Circulant Matrix: Algorithm and Theory

Rong Yin, Yong Liu, Weiping Wang, and Dan Meng

***Abstract*— Kernel ridge regression (KRR) is a powerful method for nonparametric regression. The time and space complexity of computing the KRR estimate directly are $\mathcal{O}(n^3)$ and $\mathcal{O}(n^2)$, respectively, which are prohibitive for large-scale data sets, where n is the number of data. In this article, we propose a novel random sketch technique based on the circulant matrix that achieves savings in storage space and accelerates the solution of the KRR approximation. The circulant matrix has the following advantages: It can save time complexity by using the fast Fourier transform (FFT) to compute the product of matrix and vector, its space complexity is linear, and the circulant matrix, whose entries in the first column are independent of each other and obey the Gaussian distribution, is almost as effective as the i.i.d. Gaussian random matrix for approximating KRR. Combining the characteristics of the circulant matrix and our careful design, theoretical analysis and experimental results demonstrate that our proposed sketch method, making the estimate kernel methods scalable and practical for large-scale data problems, outperforms the state-of-the-art KRR estimates in time complexity while retaining similar accuracies. Meanwhile, our sketch method provides the theoretical bound that keeps the optimal convergence rate for approximating KRR.**

***Index Terms*— Circulant matrix, kernel ridge regression (KRR), large scale, random sketch.**

I. Introduction

KERNEL ridge regression (KRR) [1]–[7] plays an important role in many scientific and engineering problems [8]–[12] and has achieved remarkable results in face images [13], [14], characterizing data [15] and so on. The computation of KRR, predicting a dependent variable based on observing an independent variable over reproducing kernel Hilbert spaces (RKHSs) [16], is equivalent to solving a linear system. However, the time and space complexity of this method is prohibitive with large-scale data sets.

Manuscript received November 7, 2018; revised March 19, 2019 and June 22, 2019; accepted September 25, 2019. This work was supported in part by the National Natural Science Foundation of China under Grant 61703396 and Grant 61673293, in part by the CCF-Tencent Open Fund, in part by the Youth Innovation Promotion Association CAS, in part by the Excellent Talent Introduction of the Institute of Information Engineering of CAS under Grant Y7Z0111107, and in part by the Beijing Municipal Science and Technology Project under Grant Z191100007119002. *(Corresponding author: Yong Liu.)*

R. Yin and Y. Liu are with the Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China, and also with the School of Cyber Security, University of Chinese Academy of Sciences, Beijing 100049, China (e-mail: yinrong@iie.ac.cn; liuyong@iie.ac.cn).

W. Wang and D. Meng are with the Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China.

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2019.2944959

Facing the bottlenecks of time and space consumption in KRR, more and more researchers have devoted themselves to improving them. Partitioning the input data set for KRR by divide-and-conquer rule [17], [18] and approximating KRR estimates [19]–[26] are the main solutions. In approximating KRR estimates, there are iterative methods, random feature, matrix approximation, sketch methods, and so on.

The typical representative of iterative methods is [27]–[29], which skillfully designs various iterations and greatly reduces the time consumption. However, they have their own shortcomings: The iterative methods in [28] can be stopped early, yet the condition number of $\mathbf{K}$ is too large to have a great number of iterations.

Random feature [29], [30] has a long and distinguished history, which is based on randomized construction of the approximate kernel functions. Although the optimality of the statistical minimaxity is guaranteed, they are still unable to perfectly cope with the time and space problems with the increasing number of data.

Matrix approximation has been widely applied and achieved remarkable results. The typical representative is the accepted Nyström method [31], [32], the prominent feature of which is to sample data before the large-scale matrix operations. There are many variants of the sampling process [33]–[35]. For example, Zhang and Kwok [31] proposed a "clustered Nyström method" that uses the k-means clustering centers as landmark points. However, the Nyström approximation is very sensitive to noninhomogeneity in the samples.

The main principle of the fourth method—sketch method [36]–[39]—is to sketch the input data so as to achieve the acceleration and other purposes. Compared with Nyström, the sketch is more suitable for the real characteristics of data sets. Several studies have demonstrated that the sketch method is an effective way to reduce the time complexity on the premise of guaranteeing a satisfactory accuracy. Wang *et al.* [39] addressed the statistical and optimization impacts of the classical sketch and the Hessian sketch, which are used to approximately solve the ridge regression problem instead of KRR and only for the $n \gg d$ problem. The representative sketch methods in KRR are the sub-Gaussian sketch method (the entries of the sketch matrices are i.i.d. Gaussian, i.i.d. Bernoulli, and so on) and the randomized orthogonal system (ROS) sketch method [the sketch matrices are the Hadamard matrix, the discrete Fourier transform (DFT) matrix, and so on], which have greatly reduced the time complexity [38].

As the sketch methods can effectively and succinctly reduce the complexity of time and space and better adapt

to noninhomogeneity data sets, by contrast, we confirmedly choose the sketch method to solve the KRR approximation problem. However, the existing sketch methods in KRR still have much room for improvement: They store and compute the complete and dense kernel matrices and sketch matrices, which consume a large amount of space and time.

A circulant matrix $\mathbf{C} \in \mathbb{R}^{m\times m}$ [40] is a structured matrix, which is fully determined by its first column so that we only need to store the first column to reconstruct the whole matrix, where m is the sketch dimension. The space complexity is $O(m)$ instead of $O(m^2)$. In addition, the circulant matrix can realize a matrix-vector product ($\mathbf{C} * \mathbf{v}$, $\mathbf{v} \in \mathbb{R}^m$) by the fast Fourier transform (FFT), whose time cost is $\mathcal{O}(m \log(m))$ [41]. Most importantly, the effectiveness of a circulant matrix whose entries in the first column are i.i.d. Gaussian entries is almost the same as that of an unstructured matrix with i.i.d. Gaussian entries [42]. Because of the advantages mentioned above, the circulant matrices have attracted wide attention in some fields: approximation of the kernel matrices [43], [44], kernel selection [45], [46], approximation of the kernel function [41], binary embedding [47], and so on. To the best of our knowledge, the circulant matrix has not been applied to KRR based on the random sketch, not to mention a theoretical justification. The purpose and method of using the circulant matrix in our method are different from those in the past.

To accelerate the solution and save the storage space of the KRR approximation, we propose a novel sketch technique based on the circulant matrix and provide the theoretical proofs. Primarily, we sample the data before generating the kernel matrix so as to significantly reduce the memory requirement and the subsequent computation. Second, a structured matrix—the circulant matrix—is carefully designed to constitute a sketch matrix and repeatedly used in our method, which significantly reduces the time and space complexity. Most importantly, the effectiveness of our circulant matrix is almost the same as that of an unstructured matrix with i.i.d. Gaussian entries [42]. Our main contributions are as follows.

1) We propose a novel KRR approximation algorithm and give a detailed proof, which shows our method keeps the optimal convergence rate.
2) In terms of training, the training time complexity of our method is $\mathcal{O}(nm^2)$, which improves the result of the state-of-the-art methods by a factor of $1 + m/n$.
3) In terms of testing, our prediction time complexity of $\mathcal{O}(md)$ outperforms the state-of-the-art methods for the KRR estimation by a factor of n/m, where d denotes the dimension of input data.

In Section II, we analyze and compare the representative methods in detail. The preliminaries and our method are introduced in Sections III and IV. We conscientiously give a reasonable proof in Section IX and assertorically obtain the best statistical convergence rate for approximating KRR in Section V. The specific information on comparison methods is provided in Section IV. Ultimately, some reasonable experiments are designed to verify the correctness of our method. We compare our algorithm with other classical methods on 12 real-world data sets and earnestly analyze the advantages and disadvantages of every method.

II. Related Work

In this section, we mainly introduce two parts: the methods to approximate KRR and the application of the circulant matrices in kernel methods.

In the sketch methods, there are many scalable methods for KRR. The representative is the Gauss method, which uses the fast sub-Gauss transform to accelerate the approximation of the kernel matrix [36]–[38]. The Gauss method in [38], like our method, achieves the optimal convergence rate (in [38, Th. 2]). However, compared to the Gauss method on the premise of guaranteeing the same accuracy rate, our method reduces the training time and prediction time by the factors of $1 + n/m$ and n/m, respectively. Meanwhile, our method also has an advantage in space complexity: reducing n/m factor.

Another classic sketch method is ROS, which uses an orthonormal matrix, such as a Hadamard or the DFT matrix, to perform matrix-vector product effectively [38]. The ROS method can also achieve the optimal convergence rate (in [38, Th. 2]). The corresponding sketch dimension m is $\log^4(n)$ times larger than the one in our method. Compared to ROS with the same accuracy rate, our method reduces the space, training time, and prediction time complexity by the factors of n/m, $\log^8(n) + n\log(n)/m^2$, and n/m. There are many variants of this kind of approximate methods, such as Hession sketch, classical sketch, sparse sketch, and so on. For more details, please refer to [36], [37], and [39] and their citations.

Nyström is a classical method for approximating the kernel matrix. The representative methods [31], [32] can effectively perform large-scale operations on approximate KRR, whose training time, prediction time, and space complexity can reach $\mathcal{O}(nm^2 + m^3)$, $\mathcal{O}(nd)$, and $\mathcal{O}(nm)$, respectively. Our method reduces the training time and prediction time by the factors of $1 + m/n$ and n/m, respectively, compared to them.

In the following, we introduce the application of the circulant matrices in kernel methods. The circulant matrix has the property of saving time and space complexity. One of the representative applications in kernel methods is the kernel matrices' approximation [43], [44]. Wilson and Nickisch [44] introduced a newly structured kernel interpolation (SKI) framework to produce kernel approximations. The concrete expression is as follows: $\mathbf{K} \approx \mathbf{W}\mathbf{K}_{U,U}\mathbf{W}^T = \mathbf{K}_{\mathrm{SKI}}$, where $\mathbf{K}_{\mathrm{SKI}}$ is the approximation of kernel matrix $\mathbf{K}$ and $\mathbf{W}$ is a sparse matrix. They exploit the Toeplitz structure in $\mathbf{K}_{U,U}$. In [43], they introduce an approximation of the kernel matrices by the appropriate multilevel circulant matrices.

Kernel selection [45], [46] is also a representative application. In [46], they propose a randomized kernel selection approach to select the kernel with the spectra of the specifically designed multilevel circulant matrices (MCMs). They construct the randomized MCMs to encode the kernel function and all data together with labels. And they build a one-to-one correspondence between all candidate kernel functions and the spectra of the randomized MCMs by Fourier transform. In [45], they also introduce the MCMs into the automatic kernel selection.

The third representative application is the approximation of kernel function [41], which obtains a novel random feature

TABLE I

COMPARISON OF THE CLASSICAL APPROXIMATION METHODS IN KRR ESTIMATES. THE THIRD COLUMN CORRESPONDS TO THE SPACE COMPLEXITY OF EVERY METHOD. THE FOURTH COLUMN CORRESPONDS TO THE TRAINING TIME COMPLEXITY. THE FIFTH COLUMN CORRESPONDS TO THE PREDICTION TIME COMPLEXITY. THE PREDICTION TIME IS OBTAINED UNDER THE CONDITION OF A PIECE OF PREDICTION DATA. n DENOTES THE NUMBER OF TRAINING DATA. d DENOTES THE DATA DIMENSION. m DENOTES THE SKETCH DIMENSION

Reference	Method	Space	Training Time	Prediction Time
The original	KRR	$\mathcal{O}\left(n^2\right)$	$\mathcal{O}\left(n^3\right)$	$\mathcal{O}\left(nd\right)$
[38]	Gauss	$\mathcal{O}\left(n^2\right)$	$\mathcal{O}\left(n^2m+nm^2\right)$	$\mathcal{O}\left(nd\right)$
[38]	ROS	$\mathcal{O}\left(n^2\right)$	$\mathcal{O}\left(n^2\log(n)+nm^2\log^8(n)\right)$	$\mathcal{O}\left(nd\right)$
[32], [31]	Nyström	$\mathcal{O}\left(nm\right)$	$\mathcal{O}\left(nm^2+m^3\right)$	$\mathcal{O}\left(nd\right)$
This paper	CKRR	$\mathcal{O}\left(nm\right)$	$\mathcal{O}\left(nm^2\right)$	$\mathcal{O}\left(md\right)$

mapping (SCRF). The formula of SCRF is as follows: $\mathbf{x} \mapsto \cos(\Pi\mathbf{x}+\mathbf{b})$, where $\Pi = [\mathbf{P}^{(1)}; \mathbf{P}^{(2)}; \ldots; \mathbf{P}^{(t)}]$, $\mathbf{P} = [\sigma_1\mathbf{C}_{1.}; \sigma_2\mathbf{C}_{2.}; \ldots; \sigma_m\mathbf{C}_{m.}]$, $\mathbf{C}_{m.}$ is the mth row vector of a circulant matrix, σ_m is taken as 1 or -1 with 1/2 probability, and b, t, and m are the related parameters in the algorithm.

In our method, we construct a sketch matrix $\mathbf{S} = \mathbf{DCQ}$ based on the circulant matrix ($\mathbf{D}$ is a diagonal matrix, $\mathbf{C}$ is a circulant matrix, and $\mathbf{Q}$ is used for sampling) to sketch data so as to accelerate KRR. The approach and purpose of using the circulant matrix in our method are different from the mentioned above.

By using the circulant matrix with the FFT, we can reduce the time complexity of matrix–matrix product ($\mathbf{CK}, \mathbf{C} \in \mathbb{R}^{m\times m}$, and $\mathbf{K} \in \mathbb{R}^{m\times n}$) from $\mathcal{O}(nm^2)$ to $\mathcal{O}(nm\log m)$ and save the storage space by storing only the first column of the circulant matrix to keep the whole information instead of storing the full matrix. In addition to making use of the characteristics of the circulant matrix, our ingeniously conceived method can also make tremendous progress in time and space complexity. Compared to the original KRR method, our method improves the training time complexity, prediction time complexity, and space complexity by the factors of n^2/m^2, n/m, and n/m, respectively.

Table I shows the specific information of the typically approximate KRR methods.

III. PRELIMINARIES

A. Basic Definitions and Notations

In this article, we use $\mathbf{x} = [x_1\ x_2\ldots]^T, \mathbf{y} = [y_1\ y_2\ldots]^T, \ldots$ to denote the vectors and $\mathbf{A}, \mathbf{B}, \ldots$ to denote the matrices. The vectors are column-vectors. $\|\cdot\|$ denotes the norm, for example $\|\cdot\|_{\mathcal{H}}$ denotes the Hilbert space norm. We denote the kernel by k, which is a function from $\mathcal{X}\times\mathcal{X}$ to $\mathbb{R}$. The kernel matrix is denoted by $\mathbf{K}$, i.e., $\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$. Details can be found in [48]. The RKHS is denoted by $\mathcal{H}_k$ and the associated inner product by $(\cdot,\cdot)_{\mathcal{H}_k}$. $\mathcal{N}(0,1)$ represents a normal distribution with a mean value of 0 and a variance of 1.

B. Kernel Ridge Regression

Given the data set $\{(x_i, y_i)\}_{i=1}^n$, our primary objective is to approximate the optimal regression model $f^*(x) = \mathbb{E}[\mathbf{Y}|\mathbf{X} = x]$, which reflects the relationship between $\mathbf{x}$ and $\mathbf{y}$. $\mathbb{E}$ denotes the expected value.

The original method of KRR is based on the convex program as follows:

$$f = \arg\min_{f\in\mathcal{H}}\left\{\frac{1}{2n}\sum_{i=1}^{n}(y_i - f(x_i))^2 + \lambda_n\|f\|_{\mathcal{H}}^2\right\} \tag{1}$$

where λ_n is the regularization parameter. In general, we can get the solution of optimization problem by solving the n-dimensional convex program instead of infinite dimension, as a straightforward result of the theorem in [49]. Subsequently, we can solve the quadratic program (2) to obtain the KRR estimate (3)

$$\boldsymbol{\alpha} = \arg\min_{\boldsymbol{\alpha}\in\mathbb{R}^n}\left\{\frac{1}{2}\boldsymbol{\alpha}^T\mathbf{K}^2\boldsymbol{\alpha} - \boldsymbol{\alpha}^T\frac{\mathbf{Ky}}{\sqrt{n}} + \lambda_n\boldsymbol{\alpha}^T\mathbf{K}\boldsymbol{\alpha}\right\} \tag{2}$$

$$f(\cdot) = \frac{1}{\sqrt{n}}\sum_{i=1}^{n}\boldsymbol{\alpha}_i k(\cdot, x_i) \tag{3}$$

where $\boldsymbol{\alpha} = [\boldsymbol{\alpha}_1\ldots\boldsymbol{\alpha}_n]^T$ and $\mathbf{y} \equiv [y_1\ldots y_n]^T$. The time complexity of the original KRR method is $\mathcal{O}(n^3)$ and the space complexity is $\mathcal{O}(n^2)$. In many applications, the number of data is very large so that the time and space needed are prohibitive. Faced with the difficulties, we should find some efficient methods to overcome the bottlenecks.

C. Sketched Kernel Ridge Regression

Here, we mainly introduce the general sketch method in KRR. Sketching is a powerful dimension reduction technique for accelerating numerical linear algebra primitives. In general, we carefully construct an embedding from a high-dimensional space in a low-dimensional one to accelerate the computation and save the storage space. The embedding can be implemented by a sketch matrix $\mathbf{S} \in \mathbb{R}^{m\times n}$, where the sketch dimension m satisfies $m \ll n$. Thereby, we can transform the original KRR method into the sketched kernel ridge regression (SKRR) estimate

$$\hat{\boldsymbol{\alpha}} = \arg\min_{\boldsymbol{\alpha}\in\mathbb{R}^m}\left\{\frac{1}{2}\boldsymbol{\alpha}^T(\mathbf{SK})(\mathbf{KS}^T)\boldsymbol{\alpha} - \boldsymbol{\alpha}^T\mathbf{S}\frac{\mathbf{Ky}}{\sqrt{n}} + \lambda_n\boldsymbol{\alpha}^T\mathbf{SKS}^T\boldsymbol{\alpha}\right\}. \tag{4}$$

Then, we can get the solution of the equation

$$\hat{f}(\cdot) = \frac{1}{\sqrt{n}} \sum_{i=1}^{n} (\mathbf{S}^T \hat{\boldsymbol{\alpha}})_i k(\cdot, x_i) \tag{5}$$

with

$$\hat{\boldsymbol{\alpha}} = (\mathbf{SK}^2\mathbf{S}^T + 2\lambda_n \mathbf{SKS}^T)^{-1}(\mathbf{SKy}).$$

The key to the success of the SKRR method is to construct an effective sketch matrix. Nowadays, there are many excellent ways to construct the sketch matrices, which have achieved outstanding results. In order to make the KRR algorithm serve the large-scale data sets better, we design a new sketch matrix based on the circulant matrix.

IV. KRR With Circulant Matrix

Here, we mainly introduce our approximate KRR method: the large-scale KRR algorithm with the circulant matrix (CKRR).

A. Circulant Matrix

We usually need a physical implementation to perform the linear projection of $\mathbf{Ax}$. When the dimension of the unstructured matrix $\mathbf{A}$, which is not sparse, is relatively large, the time and space cost is usually overly expensive. This motivates us to explore a structured matrix to quickly implement $\mathbf{Ax}$.

The circulant matrix, a structured matrix, is the key element of our sketch matrix, which is completely dependent on the entries in the first column. That is, the entries in the latter column are generated by moving the entries in the former column forward in a circular fashion, so we only need to know the entries in the first column [40]. Preserving the complete information of the circulant matrix only needs to store the entries in the first column; Therefore, this structured matrix can help us save a lot of storage space. The specific form of the circulant matrix is as follows:

$$\mathbf{C} = \begin{bmatrix} c_1 & c_m & c_{m-1} & & \cdots & c_2 \\ c_2 & c_1 & c_m & c_{m-1} & & \vdots \\ & c_2 & c_1 & c_m & \ddots & \\ \vdots & \ddots & \ddots & \ddots & \ddots & c_{m-1} \\ & & & \ddots & \ddots & c_m \\ c_m & c_{m-1} & \cdots & & c_2 & c_1 \end{bmatrix} \tag{6}$$

which can be shortened to

$$\mathbf{C}_{[m]} = \mathrm{cir}[c_j : j \in \{1, 2, \ldots, m\}]. \tag{7}$$

Noted that the (i, j) entry of $\mathbf{C}, \mathbf{C}_{ij}$, can be characterized as follows:

$$\mathbf{C}_{ij} = c_{(i-j) \bmod m}. \tag{8}$$

Saving storage space is not the only advantage of circulant matrices. Most importantly, when multiplied by a matrix or vector, it can greatly reduce the time consumption compared with the linear operation directly. The main reason is that the circulant matrix can be transformed into the form of the DFT [40], [41], [50]

$$\mathbf{C} = \frac{1}{m} \mathbf{G}^* \mathrm{diag}(\mathbf{Gc}) \mathbf{G} \tag{9}$$

where $\mathbf{c} = [c_1, c_2, \ldots, c_m]^T$ is the first column of $\mathbf{C}$ and $\mathbf{G} = [e^{i(2\pi/mkt)}]_{k,t=1}^{m}$. $\mathbf{G}$ is the discrete Fourier matrix of order $m (i = \sqrt{-1})$, whose conjugate transpose is $\mathbf{G}^*$. $\mathrm{diag}(\mathbf{Gc})$ represents the diagonal matrix, whose diagonal consists of the entries of the vector $\mathbf{Gc}$.

It is obvious that computing $\mathbf{Cx}$ ($\mathbf{x} \in \mathbb{R}^m$ is a vector) is equivalent to implementing $\mathbf{G}^* \mathrm{diag}(\mathbf{Gc}) \mathbf{Gx}$, which can be realized efficiently via the FFT algorithm [51], [52]. The time complexity is $m \log(m)$ instead of m^2.

In addition to the two characteristics of saving time and space, the circulant matrix, whose entries in the first column are independent of each other and obey the Gaussian distribution, is almost as effective as the i.i.d. Gaussian random matrix [42].

B. Our Algorithm

As described above, constructing the sketch matrix is an important part of the sketch method. We carefully construct our embedding to save the time and space complexity.

The form of our sketch matrix $\mathbf{S} \in \mathbb{R}^{m \times n}$ in (4) is as follows:

$$\mathbf{S} = \frac{1}{\sqrt{m}} \mathbf{DCQ}. \tag{10}$$

$\mathbf{D} \in \mathbb{R}^{m \times m}$ is a random diagonal matrix and the diagonal entries are i.i.d. Rademacher variables. That is, the value is $+1$ or -1 with the probability of 1/2. m is the sketch dimension. $\mathbf{C} \in \mathbb{R}^{m \times m}$ is a circulant matrix with the first column satisfying $\mathbf{c}_1 \sim \mathcal{N}(0, 1)$. $\mathbf{Q} \in \mathbb{R}^{m \times n}$ is a sampling matrix, which consists of a random subset of m rows sampled uniformly from the $n \times n$ identity matrix without replacement.

Algorithm 1 Sketch KRR Using Circulant Matrix (CKRR)

Input: Data set $\{(x_i, y_i)\}_{i=1}^{n}$.
Output: $\hat{\boldsymbol{\alpha}}$.

1: Initialize: kernel parameters, sketch dimension m, regularization parameter $\lambda_n \geq \frac{\sqrt{\log(n)}}{2n}$,
2: Construct a random diagonal matrix $\mathbf{D} \in \mathbb{R}^{m \times m}$ whose diagonal entries are i.i.d. Rademacher variables,
3: Construct a circulant matrix $\mathbf{C} \in \mathbb{R}^{m \times m}$ whose entries in first column obey the standard normal distribution,
4: Sample m data according to matrix $\mathbf{Q}$ in (10) and construct a variant kernel matrix $\mathbf{K}' \in \mathbb{R}^{m \times n}$,
5: Compute $\mathbf{SK} \in \mathbb{R}^{m \times n}$ with FFT, namely $\mathbf{SK} = \mathbf{DCK}'$,
6: Compute $\hat{\boldsymbol{\alpha}} = \left((\mathbf{SK})(\mathbf{SK})^T + 2\lambda_n \mathbf{S}(\mathbf{SK})^T\right)^{-1} (\mathbf{SK})\mathbf{y}$.

The detailed flow of our method CKRR is summarized in Algorithm 1. Before a large-scale data set is put into the mass operations, we first sample the data set by $\mathbf{Q}$ and then generate the kernel matrix whose space complexity is nm, $(m \ll n)$, just as step 4, instead of directly generating the kernel matrix with the whole data set whose space complexity is n^2. During

solving $\mathbf{SK}$ and $\mathbf{S}(\mathbf{SK})^T$ in steps 5 and 6, the FFT can be used repeatedly. These techniques can greatly reduce the memory and time consumption of our algorithm.

C. Complexity Analysis

Our sketch method can quickly and simply reduce the dimension of the kernel matrix, which brings huge savings in time and space resources. In the following, we genuinely give a detailed introduction to the time and space complexity of our method from three aspects.

1) Space Complexity: Due to sampling the data set first, we can construct a smaller kernel matrix whose space complexity is $\mathcal{O}(nm)$ instead of $\mathcal{O}(n^2)$ in KRR and other methods. This plays a leading role in the consumption of memory. Meanwhile, the circulant matrix can be obtained by only storing the first column, which consumes the space complexity $\mathcal{O}(m)$ instead of $\mathcal{O}(m^2)$ in an unstructured matrix. Taking into account the fragmented memory footprint, we can finally abbreviate our space complexity to $\mathcal{O}(mn)$.

2) Training Time Complexity: Because of the use of the structured matrix—circulant matrix, we could realize the calculation of $\mathbf{Su}$ (vector $\mathbf{u} \in \mathbb{R}^n$) efficiently via the FFT algorithm, and the time complexity is $\mathcal{O}(m \log m)$ as opposed to $\mathcal{O}(nm)$ the time required for the same operation with the unstructured Gaussian sketch in [38]. For details, see [53] and [54]. Therefore, the time complexity for solving the sketch matrix–kernel matrix product ($\mathbf{SK}$) in our method is only $\mathcal{O}(nm \log m)$.

For computing

$$\hat{\boldsymbol{\alpha}} = (\mathbf{SKKS}^T + 2\lambda_n \mathbf{SKS}^T)^{-1}\mathbf{SKy}$$

$\mathbf{SKKS}^T$ can be transformed into the form of $(\mathbf{SK}) \cdot (\mathbf{SK})^T$, the time complexity of which is $\mathcal{O}(nm^2 + nm \log(m))$. Meanwhile, the matrix–matrix product in our method can use the parallel matrix operations, which can save time at a certain multiple. $\mathbf{SKS}^T$ can be transformed into the form of $\mathbf{S} \cdot (\mathbf{SK})^T$, the time complexity of which is $\mathcal{O}(m^2 \log m)$ except for $\mathbf{SK}$. Therefore, the time complexity of computing $\hat{\boldsymbol{\alpha}}$ in our method is $\mathcal{O}(nm \log(m) + nm^2 + m^2 \log(m) + m^3 + mn)$, which can be simplified into $\mathcal{O}(nm^2)$. However, in the original KRR, calculating $\boldsymbol{\alpha} = (\mathbf{K} + 2\lambda_n \mathbf{I})^{-1}\mathbf{Ky}$ takes time $\mathcal{O}(n^3 + n^2)$.

3) Prediction Time Complexity: In terms of prediction, for predicting the label $\mathbf{y}$ of a piece of data, namely (5), we first sample and then calculate $\mathbf{K}_p$, whose time complexity is $\mathcal{O}(md)$. The time cost of $\mathbf{K}_p(\mathbf{S}^T\boldsymbol{\alpha})$ is $\mathcal{O}(m)$ except for $(\mathbf{S}^T\boldsymbol{\alpha})$ where $\mathbf{K}_p$ denotes the prediction kernel matrix. Ultimately, the prediction time complexity can be abbreviated as $\mathcal{O}(md)$. Yet in the original KRR, the prediction time complexity is $\mathcal{O}(nd + n)$.

V. Theoretical Analysis of CKRR

A kernel matrix can be eigendecomposed into $\mathbf{K} = \mathbf{U}\Lambda\mathbf{U}^T$, where $\mathbf{U} \in \mathbb{R}^{n\times n}$ is an orthonormal matrix of eigenvectors, and $\Lambda = \text{diag}\{\hat{\mu}_1, \ldots, \hat{\mu}_n\}$ is a diagonal matrix of eigenvalues with $\hat{\mu}_1 \geq \hat{\mu}_2 \geq \cdots \geq \hat{\mu}_n \geq 0$. In this article, the way we measure the quality of estimate $\hat{f}$ is as follows:

$$\|\hat{f} - f^*\|_n^2 = \frac{1}{n}\sum_{i=1}^{n}(\hat{f}(x_i) - f^*(x_i))^2.$$

At present, we analyze our method while applied to KRR estimates with the Gaussian kernels and polynomial kernels mainly.

Theorem 1 (Upper Bound): Given n samples $\{(x_i, y_i)\}_{i=1}^n$, f^* is the optimal regression function, and a sketch matrix $\mathbf{S} \in \mathbb{R}^{m\times n}$ is shown in (10). Let $\|f^*\|_{\mathcal{H}} \leq 1$ and $\lambda_n \geq ((\log(n))^{1/2}/2n)$.

1) If kernel function k satisfies the assumption

$$\exists \alpha > 1, \quad c > 0 : \hat{\mu}_i \leq c i^{-\alpha}$$

and sketch dimension $m \geq c_0\, n^{(1/\alpha+1)}$, the sketched regression estimate $\hat{f}$ from (5) satisfies the bound

$$\|\hat{f} - f^*\|_n^2 \leq c\left\{\lambda_n + \left(\frac{1}{n}\frac{1}{\alpha - 1}\right)^{\frac{\alpha}{\alpha+1}}\right\}$$

with probability greater than $1 - c_1 e^{-c_2 m} - c_3 e^{-c_4 n^{(1/\alpha+1)}(\alpha-1)^{(-\alpha/\alpha+1)}}$.

2) If kernel function k satisfies the assumption

$$\exists c > 0 : \hat{\mu}_i \leq c\frac{1}{i!}$$

and sketch dimension $m \geq c_0 \log(n/\theta^*)$, the sketched regression estimate $\hat{f}$ from (5) satisfies the bound

$$\|\hat{f} - f^*\|_n^2 \leq c\left\{\lambda_n + \frac{\theta^*}{n}\right\}$$

with probability greater than $1 - c_1 e^{-c_2 m} - c_3 e^{-c_4 \theta^*}$.

Here, constant c only depends on $\|f^*\|_{\mathcal{H}}$. c_0, c_1, c_2, c_3 and c_4 are universal constants. $(\hat{\mu}_i)_{i=1}^{\infty}$ are eigenvalues arranged in descending order. θ^* is a constant.

Proof: The proof will be given in Section IX. □

Remark 1: Except for extremely exceptional circumstances, almost all polynomial kernels satisfy the first assumption in this theorem, so that we can understand that the results under the first assumption are the conclusions of the polynomial kernels.

Without loss of generality, all Gaussian kernels satisfy the second assumption in this theorem. Meanwhile, some other types of kernels can also satisfy the second assumption, for example, the finite kernels.

In the theorem, we mainly discuss two types of kernels: the Gaussian kernels and polynomial kernels. Under the first assumption, namely the polynomial kernels, setting the regularization parameter $\lambda_n = (\log(n))^{1/2}/2n$ and having the sketch dimension $m = \Omega(n^{(1/\alpha+1)})$, the convergence rate of our algorithm can be represented as $\mathcal{O}(((\log(n))^{1/2})/2n + ((1/n)(1/\alpha - 1))^{(\alpha/\alpha+1)})$. For the Gauss kernels, if we set $\lambda_n = (\log(n))^{1/2}/2n$ and have the sketch dimension $m = \Omega(\log(n/\theta^*))$, the convergence rate can reach $\mathcal{O}(((\log(n)^{1/2}/2n) + (1/n))$, which is the same as the upper bound of the original KRR method mentioned in [38].

As described in [38], the lower bound of the error for any estimator is $\sup_{\|f^*\|_{\mathcal{H}}\leq 1}\mathbb{E}\|\tilde{f}-f^*\|_n^2 \geq c_l\delta_n^2$, where $c_l > 0$ is a constant. Combining this conclusion with the different application conditions, we get our lower bound.

Theorem 2 (Lower Bound): Given n samples $\{(x_i, y_i)\}_{i=1}^n$, any estimator $\tilde{f}$ has prediction error lower bounded as follows.

1) If kernel function k satisfies the assumption

$$\exists \alpha > 1, \quad c > 0 : \hat{\mu}_i \leq ci^{-\alpha}$$

we have

$$\sup_{\|f^*\|_{\mathcal{H}}\leq 1}\mathbb{E}\|\tilde{f}-f^*\|_n^2 \geq c_l\left(\frac{1}{n}\frac{1}{\alpha-1}\right)^{\frac{\alpha}{\alpha+1}}.$$

2) If kernel function k satisfies the assumption

$$\exists c > 0 : \hat{\mu}_i \leq c\frac{1}{i!}$$

we have

$$\sup_{\|f^*\|_{\mathcal{H}}\leq 1}\mathbb{E}\|\tilde{f}-f^*\|_n^2 \geq c_l\frac{\theta^*}{n}.$$

Here, $c_l > 0$ is a numerical constant.

For the polynomial kernels, the lower bound of the prediction error is $\mathcal{O}(((1/n)(1/\alpha-1))^{(\alpha/\alpha+1)})$. For the Gauss kernels, we can reach the lower bound $\mathcal{O}((1/n))$. Except for the slightly different scaling coefficients, our each lower bound is missing one item $((\log(n)/^{1/2})/2n)$ compared to the upper bound, which is a relatively small number. In summary, our sketch method provides the theoretical bound that keeps the optimal convergence rate for approximate KRR.

VI. Compared Methods

In this section, we briefly introduce the compared approximate methods in theory and analyze the time complexity, space complexity, and other information.

A. Gauss Method

The form of the Gauss method in KRR estimate is as follows:

$$\hat{f}_g(\cdot) = \frac{1}{\sqrt{n}}\sum_{i=1}^{n}(\mathbf{S}^T\hat{\boldsymbol{\alpha}}_g)k(\cdot, x_i)$$

with

$$\hat{\boldsymbol{\alpha}}_g = (\mathbf{S}\mathbf{K}^2\mathbf{S}^T + 2\lambda_n\mathbf{S}\mathbf{K}\mathbf{S}^T)^{-1}\cdot(\mathbf{S}\mathbf{K}\mathbf{y})$$

where $\mathbf{S} \in \mathbb{R}^{m\times n}$ and entries $s_{ij} \sim \mathcal{N}(0, 1/m)$ for $i = 1, \ldots, m$, and $j = 1, \ldots, n$.

The Gauss method [38] achieves the optimal convergence rate when the sketch dimension $m = \Omega(n^{(1/\alpha+1)})$ in the polynomial kernels and $m = \Omega(\log(n/\theta^*))$ in the Gauss kernels, where $\alpha > 1$ and θ^* is a constant. The training time complexity for solving $\hat{\boldsymbol{\alpha}}_g$ is $\mathcal{O}(n^2m+nm^2)$, and the prediction time complexity for solving $\hat{f}_g(\cdot)$ is $\mathcal{O}(nd)$ except for $\mathbf{S}^T\hat{\boldsymbol{\alpha}}_g$. However, this method needs to deal with the whole kernel matrix, which constrains the space complexity $\mathcal{O}(n^2)$. When the data set is very large, the requirement for time and space will be enormous.

B. ROS Method

The ROS method uses the orthogonal matrix to save the calculation time, which can utilize the FFT algorithm to accelerate the multiplication of matrix and vector effectively. The basic form is

$$\hat{f}_r(\cdot) = \frac{1}{\sqrt{n}}\sum_{i=1}^{n}(\mathbf{S}^T\hat{\boldsymbol{\alpha}}_r)k(\cdot, x_i)$$

with

$$\hat{\boldsymbol{\alpha}}_r = (\mathbf{S}\mathbf{K}^2\mathbf{S}^T + 2\lambda_n\mathbf{S}\mathbf{K}\mathbf{S}^T)^{-1}\cdot(\mathbf{S}\mathbf{K}\mathbf{y})$$

where $\mathbf{S}$ rows $s_i = ((n/m))^{1/2}\mathbf{R}\mathbf{H}^T p_i$ for $i = 1, \ldots, m$, $\mathbf{R} \in \mathbb{R}^{n\times n}$ is a random diagonal matrix, $\mathbf{H} \in \mathbb{R}^{n\times n}$ is a Hadamard matrix, and $\{p_1, \ldots, p_m\}$ represent random m rows of $n \times n$ identity matrix.

The ROS method [38] can achieve the optimal convergence rate when the sketch dimension $m = \Omega(n^{(1/\alpha+1)}\log^4(n))$ in the polynomial kernels and $m = \Omega(\log(n/\theta^*)\log^4(n))$ in the Gauss kernels. The training time complexity is $\mathcal{O}(n^2\log(n) + nm^2\log^8(n))$, the prediction time complexity for a piece of data is $\mathcal{O}(nd)$ except for $\mathbf{S}^T\hat{\boldsymbol{\alpha}}_r$, and the space complexity is $\mathcal{O}(n^2)$. Compared with the original KRR method, this method has been significantly improved in terms of the time complexity.

C. Nyström Method

Nyström is a widely accepted and used approximate method, which has various variants. Here, we use the formula

$$\hat{f}_{ny}(\cdot) = \sum_{i=1}^{n}\hat{\boldsymbol{\alpha}}_{ny}k(\cdot, x_i)$$

with

$$\hat{\boldsymbol{\alpha}}_{ny} = (\mathbf{K}_{ny} + 2\lambda\mathbf{I})^{-1}\cdot\mathbf{y}$$

where $\mathbf{K}_{ny}$ is from [32]. The time complexity for training and prediction is $\mathcal{O}(nm^2 + m^3)$ and $\mathcal{O}(nd)$, respectively. Meanwhile, the space consumption of this method is $\mathcal{O}(nm)$.

VII. Experiments

In order to evaluate the effectiveness of our method, we conduct extensive experiments on 12 common data sets in libsvm. KRR is essentially the same as the least squares support vector machines (LSSVMs), as they have the same objective function (the sum of the square loss and the regularization term) at the training stage. LSSVM has been widely used in classification problems. KRR can also be used for the classification and regression problems [29], [55]. In our experiments, we use the data sets of classification and regression to validate it. Each experiment is performed on the machine with 32 cores (2.40 GHz) and 64 GB of RAM. In order to avoid contingency, every kind of experiment is repeated 30 times and the average values are taken as the final results. The statistical significance of differences in performance between methods can be estimated by using multiple training/prediction partitions. P_i and E_i represent the prediction errors of methods P and E in partition i and $D_i = P_i - E_i, i \in \{1, \ldots, 30\}$. $\bar{D}$ and S

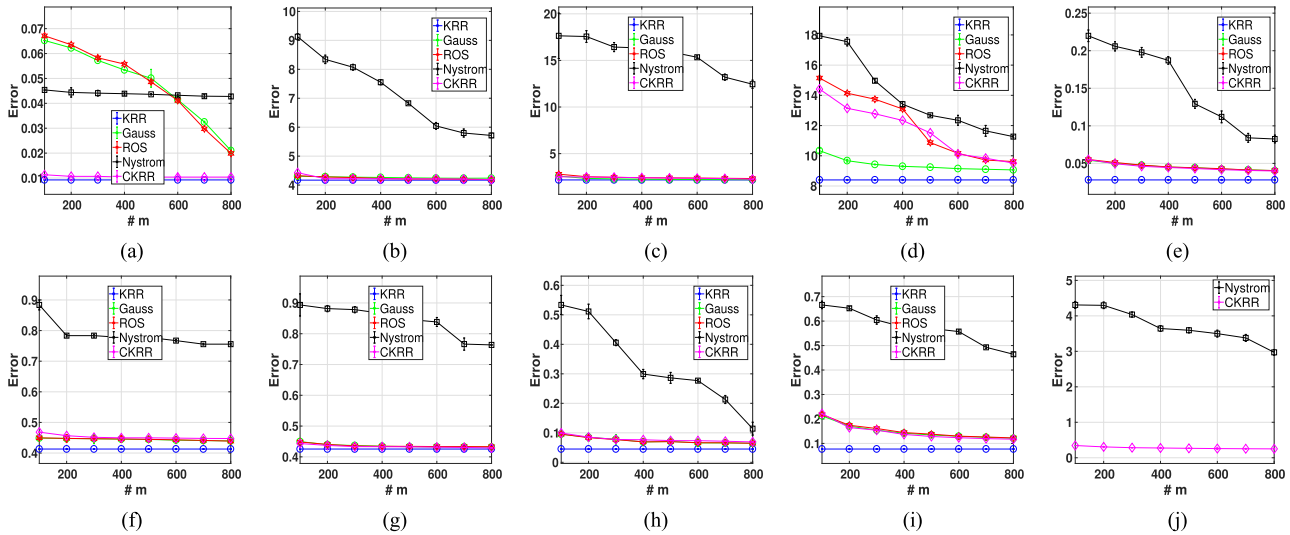

Fig. 1. Error rate of various methods on (a) space-ga, (b) abalone, (c) usps, (d) mnist.t, (e) phishing, (f) a7a, (g) a8a, (h) w7a, (i) ijcnn1, and (j) cod-rna data sets.

represent the mean and standard deviation of D_i. Under t-test, if the t-statistic $(\bar{D}/S/(30^{1/2})) > 1.699$, we have that E is significantly better than P with confidence level 95%. In the rest of this section, the statistical significance refers to a 95% level of significance.

A. Baselines and Parameter Settings

We compare our sketch method with some classical methods, which are as follows.

1) *KRR Method:* It is the original KRR method.
2) *Nyström Method:* It is now the most classical approximation method in KRR. The code is from [32].
3) *Gauss Method:* The most widely used approximation method of KRR estimates is the sub-Gaussian sketch method whose entries of the sketch matrix can be drawn from i.i.d. Gauss, i.i.d. Bernoulli, or a rescaled sphere with independent rows randomly. Here, we use the most classical Gauss sketch matrix, and the method is abbreviated as "Gauss," whose code is obtained from Yang *et al.* [38].
4) *ROS Method:* Another classical type of the sketch methods is ROS sketch methods, which use the FFT algorithm to accelerate a matrix-vector product. The matrices mainly include the orthonormal matrices of the DFT and Hadamard cases among others. Here, we choose the latest method using the Hadamard matrix as our contrast, which is abbreviated as "ROS." We get the code from Yang *et al.* [38].
5) *CKRR Method:* This represents our sketch method.

The Gauss kernel is a common and representative kernel, so we mainly do comparative experiments based on the Gauss kernel in this article. The kernel function $k(\mathbf{x}_1, \mathbf{x}_2) = e^{-(1/2h^2)(\mathbf{x}_1-\mathbf{x}_2)^2}$ is used for all types of feature representations. In the objective function (4), there are two parameters need to be determined in advance, i.e., h and λ_n. The kernel parameter h and regularization parameter λ_n in every method are chosen in $2^i, i \in \{-15, -14, \ldots, 14, 15\}$ by minimizing the prediction error via fivefold cross validation, respectively. Specifically, λ_n also satisfies $\lambda_n \geq ((\log(n)^{1/2})/2n)$ in our method.

B. Data Sets

In this article, we use 12 real-world data sets, which are collected from the libsvm website[1]: space-ga, abalone, usps, mnist.t, phishing, a7a, a8a, w7a, ijcnn1, cod-rna, skin-nonskin and YearPredictionMSD. The specific quantities and other information for every data set are shown in Table II. In our experiments, 70% of every data set is used for the training and the rest for prediction.

C. Evaluation Methodologies and Results

Under different sketch dimensions m, the errors, training time (in seconds), and prediction time (logarithmizing it) of those KRR algorithms on every data set are shown in Figs. 1–3, respectively. The formula of the error is $((\|\hat{\boldsymbol{f}} - \boldsymbol{y}\|_2^2)/n')$, where $\boldsymbol{y}$ is the vector of the real label value in prediction data, $\hat{\boldsymbol{f}}$ is the estimated label value of the corresponding data in every algorithm, and n' is the number of prediction data. We determine h and λ_n by fivefold cross validation and take KRR and our methods as example to gain deep insights on the relationship between the error rate and λ_n when $m = 200$ in Fig. 4.

From the experimental results, we have that the following holds.

1) Fig. 1 shows that our algorithm keeps a similar level of the error rate compared with the state-of-the-art KRR estimates. On the real data sets, the error rate of the Nyström method is relatively high when m is a small value. This is consistent with [38], indicating that the Nyström method is very sensitive to noninhomogeneity

[1] http://www.csie.ntu.edu.tw/∼cjlin/libsvm.

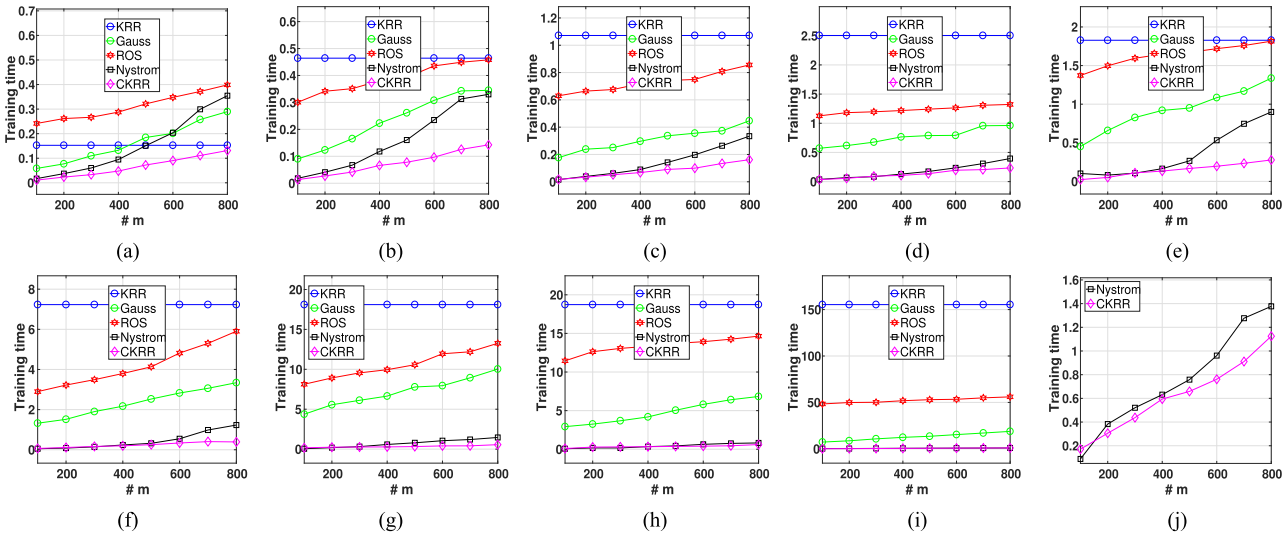

Fig. 2. Training time of various methods on (a) space-ga, (b) abalone, (c) usps, (d) mnist.t, (e) phishing, (f) a7a, (g) a8a, (h) w7a, (i) ijcnn1, and (j) cod-rna data sets.

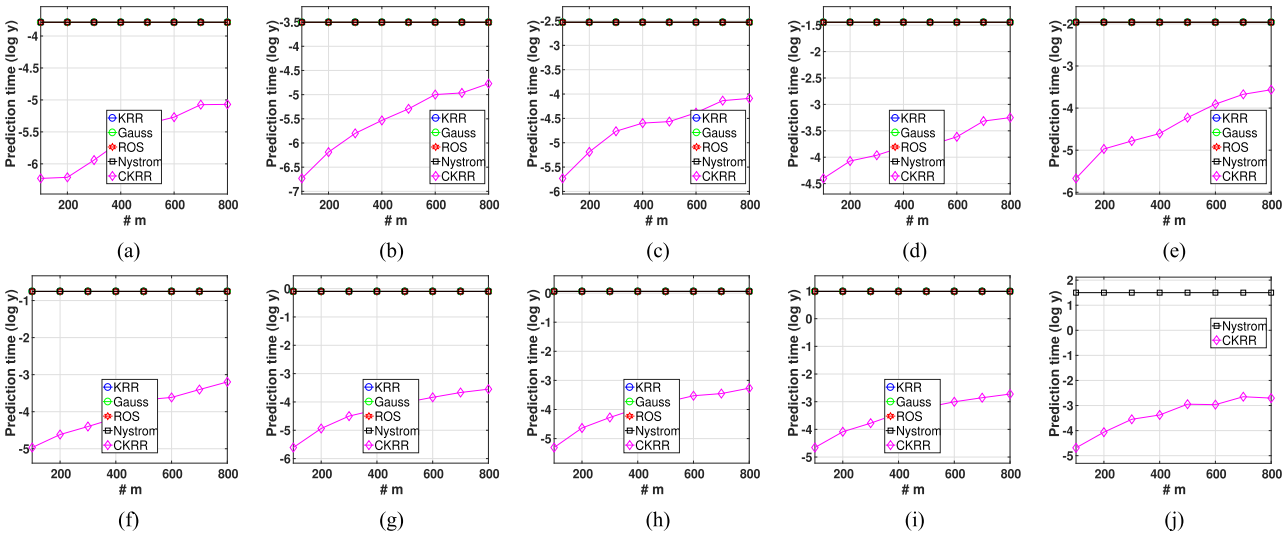

Fig. 3. Prediction time of various methods on (a) space-ga, (b) abalone, (c) usps, (d) mnist.t, (e) phishing, (f) a7a, (g) a8a, (h) w7a, (i) ijcnn1, and (j) cod-rna data sets.

in the sampling of covariates. When m is very small, the error rate of our algorithm has reached convergence. If m takes a large value, it will bring about an unnecessary waste of the computing time, memory, and other resources. For the large-scale cod-rna data set, the memory consumption of some algorithms is beyond our hardware facilities so that we do not get the related experimental data except for Nyström and CKRR. So do the following figures. Generally speaking, with the increase of m, the error rate presents a downward trend and gradually approaches the one of the KRR methods. These experimental results can exactly support our theoretical deduction.

2) Fig. 2 shows that our algorithm has a significant advantage over other approximate KRR methods in the training time. The larger the data sets, the more obvious our training time advantage is. With the increase of m, the training time of some approximate methods may exceed that of the original KRR method, such as space-ga data set in the figure. This is because generating the sketch matrix requires additional time in the approximate methods, which is included in the training time. These results are consistent with our theoretical analysis. Nyström, Gauss, ROS, and KRR algorithms are unable to get the experimental results in the larger data sets, such as skin-nonskin and YearPredictionMSD data sets. Due to the larger span of the training time, we cannot clearly identify the exact experimental values by naked eyes. We show an example of the specific values of every algorithm in Table II.
3) Fig. 3 shows that our algorithm has a significant advantage in the prediction time compared with other KRR algorithms. In the other four methods, the prediction time is not related to the sketch dimension m, but to

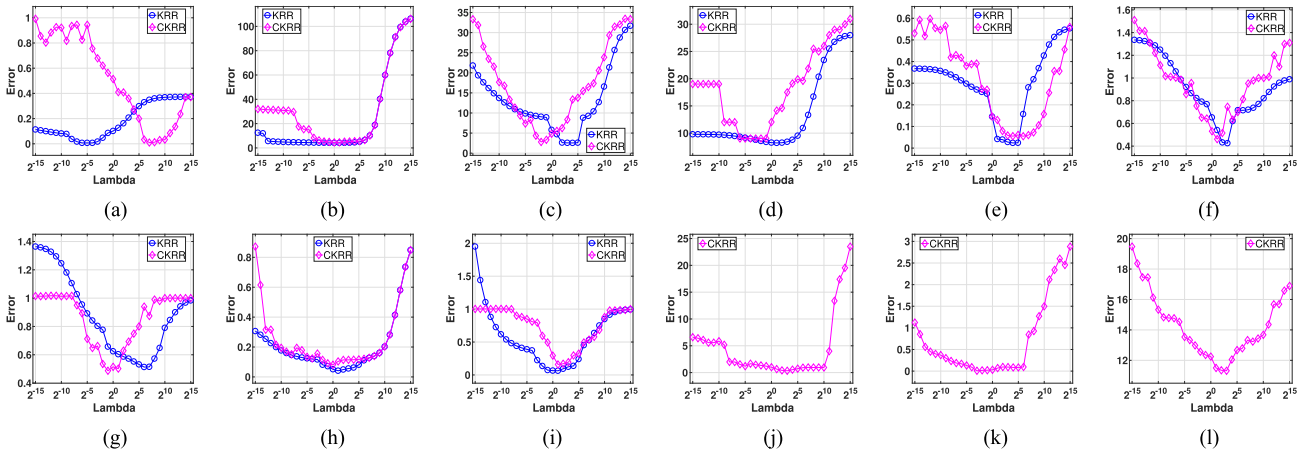

Fig. 4. Relationship between error rate and λ_n of KRR and our methods on (a) space-ga, (b) abalone, (c) usps, (d) mnist.t, (e) phishing, (f) a7a, (g) a8a, (h) w7a, (i) ijcnn1, (j) cod-rna, (k) skin-nonskin, and (l) YearPredictionMSD data sets. Setting $m = 200$.

TABLE II

COMPARISON OF AVERAGE TRAINING TIME (LEFT), PREDICTION TIME (MIDDLE) IN SECONDS, AND ERROR RATE (RIGHT) IN SOLVING KRR BETWEEN THE GAUSS METHOD, ROS METHOD, NYSTRÖM METHOD, AND OURS IN THE FOLLOWING 12 REAL-WORLD DATA SETS, WHERE $m = 1000$. m MEANS THE SKETCH DIMENSION. WE BOLD THE NUMBERS OF THE BEST METHOD AND UNDERLINE THE NUMBERS OF THE OTHER METHODS WHICH ARE NOT SIGNIFICANTLY WORSE THAN THE BEST ONE IN ERROR RATE

Dataset	Instance	Feature	Gauss			ROS			Nyström			**Ours**		
			train	pred	err	train	pred	err	train	pred	err	train	pred	err
space-ga	3,107	6	0.364	0.022	0.023	0.447	0.023	0.011	0.591	0.023	0.042	**0.150**	**0.007**	**0.010**
abalone	4,177	8	0.428	0.031	4.220	0.525	0.030	**4.182**	0.548	0.030	4.859	**0.185**	**0.010**	4.190
usps	7,291	256	0.485	0.080	**2.193**	0.917	0.080	2.257	0.503	0.080	10.10	**0.202**	**0.025**	2.214
mnist.t	10,000	780	1.025	0.236	9.017	1.357	0.236	**9.001**	0.575	0.236	9.194	**0.292**	**0.047**	9.014
phishing	11,055	68	1.460	0.141	0.039	1.890	0.141	0.039	0.967	0.142	0.055	**0.333**	**0.035**	**0.038**
a7a	16,100	123	3.930	0.471	**0.437**	6.880	0.471	0.437	1.900	0.471	0.751	**0.483**	**0.033**	0.446
a8a	22,696	123	12.30	0.906	0.441	13.20	0.906	0.440	1.610	0.906	0.749	**0.628**	**0.036**	**0.437**
w7a	24,692	300	7.810	1.060	**0.062**	15.30	1.060	0.063	0.929	1.061	0.104	**0.736**	**0.045**	0.067
ijcnn1	49,990	22	21.70	2.680	0.113	57.30	2.680	0.116	1.370	2.680	0.338	**1.180**	**0.082**	**0.111**
cod-rna	59,535	8	/	/	/	/	/	/	1.420	4.488	2.860	**1.340**	**0.089**	**0.245**
skin-nonskin	245,057	3	/	/	/	/	/	/	/	/	/	**6.270**	**0.289**	**0.001**
YearPredictionMSD	463,715	90	/	/	/	/	/	/	/	/	/	**11.20**	**0.686**	**2.723**

the number of data n. The time spent on the kernel matrix processing is also included in the prediction time. Theoretical reasoning corroborates this conclusion.

4) Fig. 4 shows that all curves are basically concave on 12 real-world data sets. Namely, when λ_n is too large or too small, the error is large. This is in line with our theoretical analysis. When KRR and our methods obtain the minimum errors, the values of λ_n are not necessarily the same. Our method of determining λ_n may not get the optimal solutions, but it is sufficient.
5) Table II shows the exact experimental values of every algorithm where $m = 1000$. When the data set is larger, some algorithms cannot get the experimental results because of the memory. Our algorithm maintains the optimal of the training and prediction time on all (12) data sets. On space-ga, phishing, a8a, ijcnn1, and cod-rna data sets, our errors are the minimum. On abalone and w7a data sets, there is no difference in errors between our method and the best of the compared methods at the 95% level of significance. On usps, mnist.t, and a7a data sets, our errors are superior to the KRR estimates except for the best method.

The experimental results show that our algorithm performs well in regression and classification.

VIII. CONCLUSION

In this article, we propose a new sketch method for approximating KRR, which opens up a new train of thought for making the KRR estimates scalable and practical in the large-scale problems. By using the circulant matrix in the sketch, we can use the FFT algorithm to reduce the time complexity during approximating KRR. It, in addition, can reduce the space complexity by only storing the first column of the circulant matrix, instead of the whole matrix, to get the complete information. Most importantly, compared with

the state-of-the-art KRR estimates, our well-designed sketch method can reduce more time complexity while achieving the satisfactory error rate, which is verified by a lot of real experiments based on 12 public data sets and rigorous theoretical analyses. Moreover, we obtain the upper and lower bounds of the convergence rate through the rigorous theoretical proof. Gaussian process (GP) regression is an important method, which has the advantage of quantifying the uncertainty and offering confidence levels. In the future, we may extend our method to GP.

IX. Proof

For smooth proof, let us introduce some basic knowledge. The statistical dimension d_n of the kernel [38] is defined as

$$d_n = \min\left\{i \in [n] : \hat{\mu}_i \le \delta_n^2\right\}$$

and if there is no such index i, setting $d_n = n$.

The critical radius δ_n, which exists and is unique for any kernel class [56], is defined to be the smallest positive solution to inequality

$$\hat{R}(\delta) \le \frac{\delta^2}{\sigma}$$

where noise variance $\sigma > 0$. σ is a simple rescaling argument of the model $y_i = f^*(x_i) + \sigma w_i$ for $i = 1, \ldots, n$ [38], [56]. $\{w_i\}_{i=1}^n$ are the noise.

The kernel complexity function $\hat{R}(\delta)$ is the form of $\hat{R}(\delta) = ((1/n)\sum_{i=1}^n \min\{\delta^2, \hat{\mu}_i\})^{1/2}$, which is generated by analyzing the local Rademacher complexity of the kernel class [56]–[59].

Before proving our theorems, we introduce two lemmas, which describe the properties of the Gauss kernels and polynomial kernels used in the theorems.

A. Proof of Lemma 1

Lemma 1: If kernel function k satisfies the assumption

$$\exists \alpha > 1, \quad c > 0 : \hat{\mu}_i \le c i^{-\alpha}$$

then there is the critical radius $\delta_n^2 = \Omega(((1/n)(1/\alpha - 1))^{(\alpha/\alpha+1)})$ and statistical dimension $d_n = \Omega(n^{(1/\alpha+1)})$, where $(\hat{\mu}_i)_{i=1}^\infty$ are eigenvalues arranged in the descending order as mentioned above.

Proof: According to the kernel complexity function, we know

$$\hat{R}(\delta) = \sqrt{\frac{1}{n}\min_{\theta \ge 0}\left(\theta\delta^2 + \sum_{i \ge \theta}\hat{\mu}_i\right)}. \tag{11}$$

Due to $\hat{\mu}_i \le c i^{-\alpha}$, we have the following inequality:

$$\sum_{i=\theta+1}^{n}\hat{\mu}_i \le c\sum_{i=\theta+1}^{n} i^{-\alpha} \le c\int_{\theta}^{\infty} x^{-\alpha}\mathrm{d}x = c\left[\frac{1}{1-\alpha}x^{1-\alpha}\right]_{\theta}^{\infty} = \frac{c}{\alpha-1}\theta^{1-\alpha}.$$

Combining (11) with the above result, we obtain

$$\min_{\theta}\left(\theta\delta^2 + \sum_{i>\theta}\hat{\mu}_i\right) \le \min_{\theta}\left(\theta\delta^2 + \frac{c}{\alpha-1}\theta^{1-\alpha}\right).$$

If setting $\theta = [(\alpha - 1c\delta^2]^{-(1/\alpha)}$, we can obtain that

$$\min_{\theta}\left(\theta\delta^2 + \sum_{i>\theta}\hat{\mu}_i\right) \le \frac{2c}{\alpha-1}\delta^{\frac{2(\alpha-1)}{\alpha}}.$$

Thus, we have

$$\hat{R}(\delta) = \sqrt{\frac{1}{n}\min_{\theta\ge 0}\left(\theta\delta^2 + \sum_{i\ge\theta}\hat{\mu}_i\right)} \le \sqrt{\frac{1}{n}\frac{2c}{\alpha-1}\delta^{\frac{2(\alpha-1)}{\alpha}}}.$$

Note that when

$$\delta \ge \left(\sigma\sqrt{\frac{1}{n}\frac{2c}{\alpha-1}}\right)^{\frac{\alpha}{\alpha+1}}$$

the following inequality holds:

$$\sqrt{\frac{1}{n}\frac{2c}{\alpha-1}\delta^{\frac{2(\alpha-1)}{\alpha}}} \le \frac{\delta^2}{\sigma}.$$

So, when let $\delta' = (\sigma((1/n)(2c/\alpha - 1))^{1/2})^{(\alpha/\alpha+1)}$, we have

$$\hat{R}(\delta') \le \sqrt{\frac{1}{n}\frac{2c}{\alpha-1}\delta'^{\frac{2(\alpha-1)}{\alpha}}} \le \frac{\delta'^2}{\sigma}.$$

Note that the critical radius is the smallest positive solution $\delta_n \ge 0$ to inequality $\hat{R}(\delta) \le (\delta^2/\sigma)$. Thus

$$\delta_n \le \delta' = \left(\sigma\sqrt{\frac{1}{n}\frac{2c}{\alpha-1}}\right)^{\frac{\alpha}{\alpha+1}}.$$

Here, we complete the proof of δ_n.

According to the definition of d_n, the solution of

$$c\frac{1}{i^\alpha} \le \left(\sigma^2\frac{1}{n}\frac{2c}{\alpha-1}\right)^{\frac{\alpha}{\alpha+1}}$$

is

$$i \ge c^{\frac{1}{\alpha}}\left(\frac{2c\sigma^2\ c}{\alpha-1}\right)^{-\frac{1}{\alpha+1}} n^{\frac{1}{\alpha+1}}.$$

Therefore, we have $d_n = \lfloor c^{(1/\alpha)}((2c\sigma^2\ c)/(\alpha-1))^{-(1/\alpha+1)}n^{(1/\alpha+1)}\rfloor$. □

B. Proof of Lemma 2

Lemma 2: If kernel function k satisfies the assumption

$$\exists c > 0 : \hat{\mu}_i \le c\frac{1}{i!}$$

then there is $\delta_n^2 = \Omega((\theta^*/n))$ and statistical dimension $d_n = \Omega(\log((n/\theta^*)))$, where θ^* is the solution of $\theta\delta^2 = c\exp(-\theta)$ with respect to θ.

Proof: The proof is similar to that of Lemma 1. Note that

$$\sum_{i=\theta+1}^{n}\hat{\mu}_i \le \sum_{i=\theta+1}^{\infty} c\frac{1}{i!} \le c\exp(-\theta).$$

Let θ^* be the solution of $\theta\delta^2 = c\exp(-\theta)$ with respect to θ. We can obtain

$$\min_{\theta}\left(\theta\delta^2 + \sum_{i>\theta}\hat{\mu}_i\right) \le (\theta\delta^2 + c\exp(-\theta)) = 2\theta^*\delta^2.$$

Thus, we have

$$\hat{R}(\delta) = \sqrt{\frac{1}{n}\min_{\theta\geq 0}\left(\theta\delta^2 + \sum_{i\geq\theta}\hat{\mu}_i\right)} \leq \sqrt{\frac{1}{n}2\theta^*\delta^2}.$$

Combining the result above, we can get the solution of $((1/n)2\theta^*\delta^2)^{1/2} \leq (\delta^2/\sigma)$ is $\delta^2 \geq (2/n)\sigma^2\theta^*$. According to the definition of the critical radius, we obtain $\delta_n^2 \leq (2/n)\sigma^2\theta^*$.

In the following, we carefully prove the value of statistical dimension $d_n = \Omega(\log((n/\theta^*)))$.

As we all know if $i \geq 1$ and $c > 0$, $ci! \geq c2^{(i-1)}$, thus $c(1/i!) \leq c(1/2^{i-1})$. The solution of $(1/2^{i-1}) \leq (2/n)\sigma^2\theta^*$ is $i \geq \log((n/\sigma^2\theta^*))$. According to the definition of statistical dimension, we have $d_n = \lfloor\log((n/\sigma^2\theta^*))\rfloor$. Here, we complete the proof of Lemma 2. □

C. Proof of Theorem 1

In order to prove our theorem 1, we first introduce two lemmas that describe the independence of the variant circulant matrix and the threshold of our sketch dimension.

Lemma 3 (Independence of Variant Circulant Matrix): The first column of a circulant matrix $\mathbf{C}$, $\mathbf{C}_{[m]} = \text{cir}[c_j : j \in \{1, 2, \ldots, m\}]$, is a random sequence with each entry drawn i.i.d. from $\mathcal{N}(0, 1/\sigma^2)$. Let the matrix $\mathbf{B} = \mathbf{DC}$, and $\mathbf{D}$ is a diagonal matrix whose diagonal entries are i.i.d. Rademacher variables. Then, the entries of matrix $\mathbf{B}$ obey $\mathcal{N}(0, 1/\sigma^2)$.

This lemma shows that the variant circulant matrix $\mathbf{B}$, coming from the circulant matrix processed by a diagonal matrix, follows the same distribution probability with random i.i.d. Gaussian matrix. However, compared with the unstructured matrix, the matrix produced in this way can use the FFT to reduce the time consumption and save storage.

Proof: The circulant matrix, whose entries in the first column are drawn i.i.d. according to some distribution probability, is a circulant random matrix. This has been proven in [41].

The value of $\mathbf{D}_{ii}$ in the diagonal matrix $\mathbf{D}$ is $+1$ or -1 with the probability of 1/2. The physical meaning of $\mathbf{DC}$ is the signs of entries in the matrix $\mathbf{C}$ that can be changed or unchanged independently with equal probability. That is, the matrix $\mathbf{D}$ eliminates the correlations between the columns of the matrix $\mathbf{C}$. The entries of $\mathbf{B}$, $\mathbf{B}_{ij} = \mathbf{D}_{ii}\mathbf{C}_{ij}$, retain Gaussianity so that $\mathbf{B}_{\cdot j}$ is also Gaussian vector with i.i.d. entries, where

$$\mathbb{E}[\mathbf{B}_{ij}] = \mathbb{E}[\mathbf{D}_{ii}\mathbf{C}_{ij}] = 0$$

and

$$\begin{aligned}\text{Var}[\mathbf{B}_{ij}] &= \text{Var}[\mathbf{D}_{ii}\mathbf{C}_{ij}] \\ &= \mathbb{E}[\mathbf{D}_{ii}^2\mathbf{C}_{ij}^2] - \mathbb{E}[\mathbf{D}_{ii}\mathbf{C}_{ij}]^2 = 1/\sigma^2\end{aligned}$$

for $i, j \in 1, \ldots, m$. Therefore, $\mathbf{B}_{ij} \sim \mathcal{N}(0, 1/\sigma^2)$. □

Lemma 4 (Sketch Dimension): Under the condition of Lemma 1: Given the sketch dimension $m \geq c_0\, n^{(1/\alpha+1)}$, the sketch matrix $\mathbf{S}$ in (10) satisfies the following inequalities:

$$\begin{aligned}\|(\mathbf{SU}_1)^T\mathbf{SU}_1 - \mathbf{I}_{d_n}\|_{op} &\leq 1/2 \\ \|\mathbf{SU}_2\Lambda_2^{1/2}\|_{op} &\leq c\left(\left(\frac{1}{n}\frac{1}{\alpha-1}\right)^{\frac{\alpha}{\alpha+1}}\right)^{1/2}\end{aligned} \tag{12}$$

with the probability of at least $c_1e^{-c_2m}$.

Under the condition of Lemma 2: Given the sketch dimension $m \geq c_0\log(n/\theta^*)$, the sketch matrix $\mathbf{S}$ in (10) satisfies the following inequalities:

$$\begin{aligned}\|(\mathbf{SU}_1)^T\mathbf{SU}_1 - \mathbf{I}_{d_n}\|_{op} &\leq 1/2 \\ \|\mathbf{SU}_2\Lambda_2^{1/2}\|_{op} &\leq c\left(\frac{\theta^*}{n}\right)^{1/2}\end{aligned} \tag{13}$$

with probability at least $c_1e^{-c_2m}$.

Here, c, c_1 and c_2 are universal constants. $\mathbf{I}$ is the identity matrix. $\Lambda_2 = \text{diag}\{\hat{\mu}_{d_n+1}, \ldots, \hat{\mu}_n\}$ belongs to the kernel matrix $\mathbf{K} = \mathbf{U}\Lambda\mathbf{U}^T$, where $\Lambda = \text{diag}\{\hat{\mu}_1, \ldots, \hat{\mu}_n\}$. $\mathbf{U}_1 \in \mathbb{R}^{n\times d_n}$ and $\mathbf{U}_2 \in \mathbb{R}^{n\times(n-d_n)}$ denote the left and right blocks of $\mathbf{U}$, respectively.

For the matrix satisfying this lemma, we call it K-satisfiable sketch matrix, which is the same as [38].

Proof: First, we prove the first inequalities in (12) and (13). Let $\mathbf{Q} = \mathbf{U}_1^T\mathbf{S}^T\mathbf{SU}_1 - \mathbf{I}_{d_n}$, where the matrix $\mathbf{U}_1 \in \mathbb{R}^{n\times d_n}$ has the orthonormal columns. Let $\{\boldsymbol{\nu}^1, \ldots, \boldsymbol{\nu}^N\}$ be 1/2-cover of the Euclidean sphere $\mathcal{S}^{d_n-1}$. By standard arguments, we can find such a set with $N \leq e^{2d_n}$ elements. Using this cover, a straightforward discretization argument yields

$$\begin{aligned}\|\mathbf{Q}\|_{op} &\leq 4\max_{j,k=1,\ldots,N}\langle\boldsymbol{\nu}^j, \mathbf{Q}\boldsymbol{\nu}^k\rangle \\ &= 4\max_{j,k=1,\ldots,N}(\tilde{\boldsymbol{\nu}})^j\{\mathbf{S}^T\mathbf{S} - \mathbf{I}_n\}\tilde{\boldsymbol{\nu}}^k\end{aligned} \tag{14}$$

where $\tilde{\boldsymbol{\nu}}^j = \mathbf{U}_1\boldsymbol{\nu}^j \in \mathcal{S}^{n-1}$, and $\tilde{Q} = \mathbf{S}^T\mathbf{S} - \mathbf{I}_n$. Standard subexponential bounds imply that

$$\mathbb{P}[(\tilde{\boldsymbol{\nu}})^j\tilde{Q}\tilde{\boldsymbol{\nu}}^k \geq 1/8] \leq c_1e^{-c_2m}$$

and consequently, by the union bound, we have

$$\mathbb{P}[\|\mathbf{Q}\|_{op} \geq 1/2] \leq c_1e^{-c_2m+4d_n} \leq c_1e^{-c_2'm}. \tag{15}$$

Second, we prove another inequalities in (12) and (13). Utilizing the $n \times n$ diagonal matrix $\bar{\Lambda} = \text{diag}(0_{d_n}, \Lambda_2)$, we have $\mathbf{SU}_2\Lambda_2^{1/2} = \mathbf{SU}\bar{\Lambda}^{1/2}$. By the definition of the matrix spectral norm, we know

$$\|\mathbf{SU}\bar{\Lambda}^{1/2}\|_{op} = \sup_{\mathbf{u}\in\mathbf{S}^{m-1}}\sup_{\boldsymbol{\nu}\in\mathcal{E}}\langle\mathbf{u}, \mathbf{S}\boldsymbol{\nu}\rangle \tag{16}$$

where $\mathcal{E} = \{\boldsymbol{\nu} \in \mathbb{R}^n | \|\mathbf{U}\bar{\Lambda}\boldsymbol{\nu}\|_2 \leq 1\}$, and $\mathbf{S}^{m-1} = \{\mathbf{u} \in R^m | \|\mathbf{u}\|_2 = 1\}$. We may choose a 1/2-cover $\{\mathbf{u}^1, \ldots, \mathbf{u}^M\}$ of the set $\mathcal{S}^{m-1}$ with $\log M \leq 2m$ elements. We then have

$$\begin{aligned}\|\mathbf{SU}\bar{\Lambda}^{1/2}\|_{op} &\leq \max_{j\in[M]}\sup_{\boldsymbol{\nu}\in\mathcal{E}}\langle\mathbf{u}^j, \mathbf{S}\boldsymbol{\nu}\rangle + \frac{1}{2}\sup_{u\in S^{d_n-1}}\sup_{\boldsymbol{\nu}\in\mathcal{E}}\langle\mathbf{u}, \mathbf{S}\boldsymbol{\nu}\rangle \\ &= \max_{j\in[M]}\sup_{\boldsymbol{\nu}\in\mathcal{E}}\langle\mathbf{u}^j, \mathbf{S}\boldsymbol{\nu}\rangle + \frac{1}{2}\|\mathbf{SU}\bar{\Lambda}^{1/2}\|_{op}\end{aligned}$$

so we can obtain that

$$\|\mathbf{SU}\bar{\Lambda}^{1/2}\|_{op} \leq 2\underbrace{\max_{j\in[M]}\sup_{\boldsymbol{\nu}\in\mathcal{E}}\langle\mathbf{u}^j, \mathbf{S}\boldsymbol{\nu}\rangle}_{\tilde{Z}}. \tag{17}$$

For each fixed $\mathbf{u}^j \in \mathcal{S}^{d_n-1}$, consider the random variable $Z^j = \sup_{\boldsymbol{\nu}\in\mathcal{E}}\langle\mathbf{u}^j, \mathbf{S}\boldsymbol{\nu}\rangle$. It is equal in distribution to the random

variable $V(\mathbf{g}) = (1/\sqrt{m})\sup_{\boldsymbol{\nu}\in\mathcal{E}}\langle \mathbf{g}, \boldsymbol{\nu}\rangle$, where $\mathbf{g} \in \mathbb{R}^n$ is a standard Gaussian vector. For $\mathbf{g}, \mathbf{g}' \in \mathbb{R}^n$, we have

$$\begin{aligned} |V(\mathbf{g}) - V(\mathbf{g}')| &\le \frac{2}{\sqrt{m}} \sup_{\boldsymbol{\nu}\in\mathcal{E}} |\langle \mathbf{g} - \mathbf{g}', \boldsymbol{\nu}\rangle| \\ &\le \frac{2\|\Lambda_2^{1/2}\|_{op}}{\sqrt{m}} \|\mathbf{g} - \mathbf{g}'\|_2 \le \frac{2\delta_n}{\sqrt{m}} \|\mathbf{g} - \mathbf{g}'\|_2 \end{aligned} \tag{18}$$

where we have used the fact that $\hat{\mu}_j \le \delta_n^2$ for all $j \ge d_n + 1$. By Lemma 3, we know our vector $\sqrt{m}s_i$ in (10) is a standard Gaussian vector. Consequently, by the concentration of measure for Lipschitz functions of Gaussian random variables, we have

$$\mathbb{P}[V(\mathbf{g}) \ge \mathbb{E}[V(\mathbf{g})] + t] \le e^{-\frac{mt^2}{8\delta_n^2}}. \tag{19}$$

Combining $m \ge n\delta_n^2$ and $((\sum_{j+d_n+1}^n \mu_j/n)^{1/2}) \le \delta_n^2$ and turning to the expectation, we can get

$$\begin{aligned} \mathbb{E}[V(\mathbf{g})] &= \frac{2}{\sqrt{m}} \mathbb{E}\|\Lambda_2^{1/2}\mathbf{g}\|_2 \le 2\sqrt{\frac{\sum_{j=d_n+1}^n \mu_j}{m}} \\ &= 2\sqrt{\frac{n}{m}}\sqrt{\frac{\sum_{j=d_n+1}^n \mu_j}{n}} \le 2\delta_n. \end{aligned} \tag{20}$$

Combining the pieces, we have shown that $P[Z^j \ge c_0(1+\varepsilon)\delta_n] \le e^{-c_2 m}$, for each $j = 1, \ldots, M$. Finally, setting $t = c\delta_n$ in the tail bound (19) for a constant $c \ge 1$ is large enough to ensure that $(c_2 m/8) \ge 2\log M$. Taking the union bound over all $j \in [M]$ yields

$$P[\|\mathbf{S}\mathbf{U}\bar{\Lambda}^{1/2}\|_{op} \ge 8c\delta_n] \le c_1 e^{-\frac{c_2 m}{8} + \log m} \le c_1 e^{-c_2' m}. \tag{21}$$

Combining the value of δ_n mentioned above in Lemmas 1 and 2, we complete the proof. □

Combined with the two lemmas and the conclusions in [38], in the following, we complete the whole proof of our Theorem 1.

Proof: We can transform our goal into the following form:

$$\|\hat{f} - f^*\|_n^2 \le 2\Big(\underbrace{\|f^\dagger - f^*\|_n^2}_{(1)} + \underbrace{\|f^\dagger - \hat{f}\|_n^2}_{(2)}\Big)$$

where the fitted function $f^\dagger$ is the solution in the case of zero noise. In the following, $\|\hat{f} - f^*\|_n^2$ is divided into two parts: approximation error (1) and estimation error (2) to prove our upper bound.

In formula (1), combining our Lemmas 2 and 4 in [38], we obtain the following information. Under the condition of Lemma 1, $\|f^\dagger - f^*\|_n^2 \le c\{\lambda_n + ((1/n)(1/\alpha - 1))^{(\alpha/\alpha+1)}\}$ with the probability of at least $c_1 e^{-c_2 m}$. Under the condition of Lemma 2, $\|f^\dagger - f^*\|_n^2 \le c\{\lambda_n + (\theta^*/n)\}$ with the probability of at least $c_1 e^{-c_2 m}$.

In formula (2), combining our Lemmas 1 and 4 in [38], we obtain under the condition of Lemma 1, $\|f^\dagger - \hat{f}\|_n^2 \le c((1/n)(1/\alpha - 1))^{(\alpha/\alpha+1)}$ with probability at least $1 - c_1 e^{-c_2 n((1/n)(1/\alpha-1))^{(\alpha/\alpha+1)}}$, and under the condition of Lemma 2, $\|f^\dagger - \hat{f}\|_n^2 \le c(\theta^*/n)$ with the probability of at least $1 - c_1 e^{-c_2\theta^*}$.

Combining the above conclusions, we complete the proof of this theorem. □

REFERENCES

[1] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York, NY, USA: Springer, 2009.

[2] V. Vovk, *Kernel Ridge Regression*. Berlin, Germany: Springer, 2013.

[3] Y. Liu, H. Lin, L. Ding, W. Wang, and S. Liao, "Fast cross-validation," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, 2018, pp. 2497–2503.

[4] Y. Liu, S. Liao, H. Lin, Y. Yue, and W. Wang, "Infinite kernel learning: Generalization bounds and algorithms," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 2280–2286.

[5] Y. Liu and S. Liao, "Eigenvalues ratio for kernel selection of kernel methods," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 2814–2820.

[6] Y. Liu, S. Jiang, and S. Liao, "Efficient approximation of cross-validation for kernel methods using Bouligand influence function," in *Proc. 31st Int. Conf. Mach. Learn.*, vol. 32, 2014, pp. 324–332.

[7] J. Li, Y. Liu, R. Yin, H. Zhang, L. Ding, and W. Wang, "Multi-class learning: From theory to algorithm," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 1586–1595.

[8] R. Boloix-Tortosa, J. J. Murillo-Fuentes, F. J. Payán-Somet, and F. Pérez-Cruz, "Complex Gaussian processes for regression," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 11, pp. 5499–5511, Nov. 2018.

[9] X.-Y. Zhang, L. Wang, S. Xiang, and C.-L. Liu, "Retargeted least squares regression algorithm," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 9, pp. 2206–2213, Sep. 2014.

[10] D. You, C. F. Benitez-Quiroz, and A. M. Martinez, "Multiobjective optimization for model selection in kernel methods in regression," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 10, pp. 1879–1893, Oct. 2014.

[11] F. A. Tobar, S.-Y. Kung, and D. P. Mandic, "Multikernel least mean square algorithm," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 2, pp. 265–277, Feb. 2014.

[12] S. V. Vaerenbergh, M. Lazaro-Gredilla, and I. Santamaria, "Kernel recursive least-squares tracker for time-varying regression," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 8, pp. 1313–1326, Aug. 2012.

[13] S. An, W. Liu, and S. Venkatesh, "Face recognition using kernel ridge regression," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, Jun. 2007, pp. 1–7.

[14] B. G. V. Kumar and R. Aravind, "Face hallucination using OLPP and kernel ridge regression," in *Proc. 15th IEEE Int. Conf. Image Process.*, Oct. 2008, pp. 353–356.

[15] G. C. Cawley, N. L. C. Talbot, R. J. Foxall, S. R. Dorling, and D. P. Mandic, "Heteroscedastic kernel ridge regression," *Neurocomputing*, vol. 57, pp. 105–124, Mar. 2004.

[16] E. J. Wegman, *Reproducing Kernel Hilbert Spaces*. Hoboken, NJ, USA: Wiley, 2004.

[17] E. Liu, Y. Song, and J. Liang, "An accelerator for kernel ridge regression algorithms based on data partition," *J. Univ. Sci. Technol. China*, vol. 48, no. 4, pp. 284–289, 2018.

[18] Y. Yang, J. Demmel, C.-J. Hsieh, and R. Vuduc, "Accurate, fast and scalable kernel ridge regression on parallel and distributed systems," in *Proc. Int. Conf. Supercomput.*, 2018, pp. 307–317.

[19] T. Gasser and H.-G. Müller, "Kernel estimation of regression functions," in *Smoothing Techn. Curve Estimation*. Berlin, Germany: Springer, 1979, pp. 23–68.

[20] H. Wandl, *On Kernel Estimation of Regression Functions*. Akademie der Wissenschaften der DDR Zentralinstitut für Mathematik und Mechanik, 1980.

[21] L. Xu and J. Jia, "Two-phase kernel estimation for robust motion deblurring," in *Proc. Eur. Conf. Comput. Vis.* Berlin, Germany: Springer, 2010, pp. 157–170.

[22] B. E. Hansen, "Uniform convergence rates for kernel estimation with dependent data," *Econ. Theory*, vol. 24, no. 3, pp. 726–748, 2008.

[23] W. K. Newey, "Kernel estimation of partial means and a general variance estimator," *Econ. Theory*, vol. 10, no. 2, pp. 233–253, 1994.

[24] L. Ding and S. Liao, "Approximate consistency: Towards foundations of approximate kernel selection," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases*, 2014, pp. 354–369.

[25] L. Ding *et al.*, "Approximate kernel selection with strong approximate consistency," in *Proc. 33rd AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 3462–3469.

[26] L. Ding and S. Liao, "Approximate model selection for large scale LSSVM," in *Proc. Asian Conf. Mach. Learn.*, vol. 20, 2011, pp. 165–180.

[27] S. O. Cheng, X. Mary, S. Canu, and A. J. Smola, "Learning with non-positive kernels," in *Proc. 21st Int. Conf. Mach. Learn.*, 2004, p. 81.

[28] G. Blanchard and N. Krämer, "Optimal learning rates for kernel conjugate gradient regression," in *Proc. Adv. Neural Inf. Process. Syst.*, 2010, pp. 226–234.
[29] H. Avron, K. L. Clarkson, and D. P. Woodruff, "Faster kernel ridge regression using sketching and preconditioning," *SIAM J. Matrix Anal. Appl.*, vol. 38, no. 4, pp. 1116–1138, 2017.
[30] A. Rahimi and B. Recht, "Random features for large-scale kernel machines," in *Proc. Adv. Neural Inf. Process. Syst.*, 2008, pp. 1177–1184.
[31] K. Zhang and J. T. Kwok, "Clustered Nyström method for large scale manifold learning and dimension reduction," *IEEE Trans. Neural Netw.*, vol. 21, no. 10, pp. 1576–1587, Oct. 2010.
[32] M. Li, J. T. Kwok, and B.-L. Lu, "Making large-scale Nyström approximation possible," in *Proc. 27th Int. Conf. Mach. Learn.*, 2010, pp. 631–638.
[33] C. K. I. Williams and M. Seeger, "Using the Nyström method to speed up kernel machines," in *Proc. Adv. Neural Inf. Process. Syst.*, 2001, pp. 682–688.
[34] S. Kumar, M. Mohri, and A. Talwalkar, "Sampling Methods for the Nyström Method," *J. Mach. Learn. Res.*, vol. 13, pp. 981–1006, Apr. 2012.
[35] A. Gittens and M. W. Mahoney, "Revisiting the Nyström method for improved large-scale machine learning," *J. Mach. Learn. Res.*, vol. 17, pp. 1–65, Apr. 2016.
[36] V. C. Raykar and R. Duraiswami, "Fast large scale Gaussian process regression using approximate matrix-vector products," in *Proc. Learn. Workshop*, 2007, pp. 1–8.
[37] V. I. Morariu, B. V. Srinivasan, V. C. Raykar, R. Duraiswami, and L. S. Davis, "Automatic online tuning for fast Gaussian summation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2009, pp. 1113–1120.
[38] Y. Yang, M. Pilanci, and M. J. Wainwright, "Randomized sketches for kernels: Fast and optimal nonparametric regression," *Ann. Statist.*, vol. 45, no. 3, pp. 991–1023, 2017.
[39] S. Wang, A. Gittens, and M. W. Mahoney, "Sketched ridge regression: Optimization perspective, statistical perspective, and model averaging," *J. Mach. Learn. Res.*, vol. 18, no. 1, pp. 8039–8088, 2017.
[40] P. J. Davis, *Circulant Matrices*. Providence, RI, USA: American Mathematical Society, 2012.
[41] C. Feng, Q. Hu, and S. Liao, "Random feature mapping with signed circulant matrix projection," in *Proc. 24th Int. Joint Conf. Artif. Intell.*, 2015, pp. 3490–3496.
[42] W. Yin, S. Morgan, J. Yang, and Y. Zhang, "Practical compressive sensing with toeplitz and circulant matrices," *Proc. SPIE, Int. Soc. Opt. Eng.*, vol. 7744, pp. 182–191, Jul. 2010.
[43] G. Song and Y. Xu, "Approximation of high-dimensional kernel matrices by multilevel circulant matrices," *J. Complex.*, vol. 26, no. 4, pp. 375–405, 2010.
[44] A. G. Wilson and H. Nickisch, "Kernel interpolation for scalable structured Gaussian processes (KISS-GP)," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1775–1784.
[45] L. Ding and S. Liao, "An approximate approach to automatic kernel selection," *IEEE Trans. Cybern.*, vol. 47, no. 3, pp. 554–565, Mar. 2017.
[46] L. Ding, S. Liao, Y. Liu, P. Yang, and X. Gao, "Randomized kernel selection with spectra of multilevel circulant matrices," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 2910–2917.
[47] F. X. Yu, S. Kumar, Y. Gong, and S.-F. Chang, "Circulant binary embedding," in *Proc. 31st Int. Conf. Mach. Learn.*, vol. 32, 2014, pp. II-946–II-954.
[48] C. Saunders, A. Gammerman, and V. Vovk, "Ridge regression learning algorithm in dual variables," in *Proc. 15th Int. Conf. Mach. Learn.*, Madison, WI, USA, Jul. 1998, pp. 515–521.
[49] G. Kimeldorf and G. Wahba, "Some results on Tchebycheffian spline functions," *J. Math. Anal. Appl.*, vol. 33, no. 1, pp. 82–95, 1971.
[50] E. E. Tyrtyshnikov, "A unifying approach to some old and new theorems on distribution and clustering," *Linear Algebra Appl.*, vol. 232, pp. 1–43, Jan. 1996.
[51] P. J. Davis, *Circulant Matrices*. Providence, RI, USA: American Mathematical Society, 2013.
[52] R. H. Chan and G. Strang, "Toeplitz equations by conjugate gradients with circulant preconditioner," *SIAM J. Sci. Stat. Comput.*, vol. 10, no. 1, pp. 104–119, 2006.
[53] N. Ailon and E. Liberty, "Fast dimension reduction using Rademacher series on dual BCH codes," *Discrete Comput. Geometry*, vol. 42, no. 4, p. 615, 2009.
[54] M. Pilanci and M. J. Wainwright, "Randomized sketches of convex programs with sharp guarantees," *IEEE Trans. Inf. Theory*, vol. 61, no. 9, pp. 5096–5115, Sep. 2015.
[55] A. Alaoui and M. W. Mahoney, "Fast randomized kernel ridge regression with statistical guarantees," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 775–783.
[56] P. L. Bartlett, O. Bousquet, and S. Mendelson, "Local Rademacher complexities," *Ann. Statist.*, vol. 33, no. 4, pp. 1497–1537, 2005.
[57] V. Koltchinskii, "Local Rademacher complexities and oracle inequalities in risk minimization," *Ann. Statist.*, vol. 34, no. 6, pp. 2593–2656, 2006.
[58] S. Mendelson, "Geometric parameters of kernel machines," in *Proc. Int. Conf. Comput. Learn. Theory*. Berlin, Germany: Springer, 2002, pp. 29–43.
[59] G. Raskutti, M. J. Wainwright, and B. Yu, "Minimax-optimal rates for sparse additive models over kernel classes via convex programming," *J. Mach. Learn. Res.*, vol. 13, pp. 389–427, Feb. 2012.

Rong Yin received the master's degree from the Harbin Institute of Technology, Harbin, China, in 2016. She is currently pursuing the Ph.D. degree with the Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China, and also with the School of Cyber Security, University of Chinese Academy of Sciences, Beijing.

Her current research interests include machine learning, optimization with respect to kernel learning, and large-scale kernel methods.

Yong Liu was born in 1986. He received the Ph.D. degree in computer science from Tianjin University, Tianjin, China, in 2016.

He is currently an Assistant Professor with the Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China, and also with the School of Cyber Security, University of Chinese Academy of Sciences, Beijing. His current research interests include large-scale kernel methods, large-scale model selection, and machine learning.

Weiping Wang received the Ph.D. degree in computer science from the Harbin Institute of Technology, Harbin, China, in 2006.

He is currently a Professor with the Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China. His research interests include big data, data security, database, and storage systems. He has more than 70 publications in major journals and international conferences.

Dan Meng received the Ph.D. degree in computer science from the Harbin Institute of Technology, Harbin, China, in 1995.

He is currently a Professor with the Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China. He has more than 100 publications in major journals and international conferences. His current research interests include machine learning, data security, database, and storage systems.