Random Features for Kernel Approximation: A Survey on Algorithms, Theory, and Beyond

Fanghui Liu, Xiaolin Huang, Yudong Chen, Johan A.K. Suykens

Abstract—The class of random features is one of the most popular techniques to speed up kernel methods in large-scale problems. Related works have been recognized by the NeurIPS Test-of-Time award in 2017 and the ICML Best Paper Finalist in 2019. The body of work on random features has grown rapidly, and hence it is desirable to have a comprehensive overview on this topic explaining the connections among various algorithms and theoretical results. In this survey, we systematically review the work on random features from the past ten years. First, the motivations, characteristics and contributions of representative random features based algorithms are summarized according to their sampling schemes, learning procedures, variance reduction properties and how they exploit training data. Second, we review theoretical results that center around the following key question: how many random features are needed to ensure a high approximation quality or no loss in the empirical/expected risks of the learned estimator. Third, we provide a comprehensive evaluation of popular random features based algorithms on several large-scale benchmark datasets and discuss their approximation quality and prediction performance for classification. Last, we discuss the relationship between random features and modern over-parameterized deep neural networks (DNNs), including the use of high dimensional random features in the analysis of DNNs as well as the gaps between current theoretical and empirical results. This survey may serve as a gentle introduction to this topic, and as a users' guide for practitioners interested in applying the representative algorithms and understanding theoretical results under various technical assumptions. We hope that this survey will facilitate discussion on the open problems in this topic, and more importantly, shed light on future research directions.

Index Terms—random features, kernel approximation, generalization properties, over-parameterized models

1 Introduction

Kernel methods [1], [2], [3] are one of the most powerful techniques for nonlinear statistical learning problems with a wide range of successful applications. Let $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X} \subseteq \mathbb{R}^d$ be two samples and $\phi : \mathcal{X} \to \mathcal{H}$ be a nonlinear feature map transforming each element in $\mathcal{X}$ into a reproducing kernel Hilbert space (RKHS) $\mathcal{H}$, in which the inner product between $\phi(\boldsymbol{x})$ and $\phi(\boldsymbol{x}')$ endowed by $\mathcal{H}$ can be computed using a kernel function $k(\cdot,\cdot) : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ as $\langle \phi(\boldsymbol{x}), \phi(\boldsymbol{x}') \rangle_{\mathcal{H}} = k(\boldsymbol{x}, \boldsymbol{x}')$. In practice, the kernel function k is directly given to obtain the inner product $\langle \phi(\boldsymbol{x}), \phi(\boldsymbol{x}') \rangle_{\mathcal{H}}$ instead of finding the explicit expression of ϕ, which is known as the *kernel trick*. Benefiting from this scheme, kernel methods are effective for learning nonlinear structures but often suffer from scalability issues in large-scale problems due to high space and time complexities. For instance, given n samples in the original d-dimensional space $\mathcal{X}$, kernel ridge regression (KRR) requires $\mathcal{O}(n^3)$ training time and $\mathcal{O}(n^2)$ space to store the kernel matrix, which is often computationally infeasible when n is large.

To overcome the poor scalability of kernel methods, kernel approximation is an effective technique by constructing an explicit mapping $\Psi : \mathbb{R}^d \to \mathbb{R}^s$ such that $k(\boldsymbol{x}, \boldsymbol{y}) \approx \Psi(\boldsymbol{x})^\top \Psi(\boldsymbol{y})$. By doing so, an efficient linear model can be well learned in the transformed space with $\mathcal{O}(ns^2)$ time and $\mathcal{O}(ns)$ memory while retaining the expressive power of nonlinear methods. A series of kernel approximation algorithms have been developed in the past years, including divide-and-conquer approaches [4], [5], [6], greedy basis selection techniques [7] and Nyström methods [8]. These methods provide a data dependent vector representation of the kernel. Random Fourier features (RFF) [9], on the other hand, is a typical data-independent technique to approximate the kernel function using an explicit feature mapping. This survey focuses on RFF and its variants for kernel approximation. RFF applies in particular to shift-invariant (also called "stationary") kernels that satisfy $k(\boldsymbol{x}, \boldsymbol{x}') = k(\boldsymbol{x} - \boldsymbol{x}')$. By virtue of the correspondence between a shift-invariant kernel and its Fourier spectral density, the kernel can be approximated by $k(\boldsymbol{x}, \boldsymbol{x}') \approx \langle \varphi(\boldsymbol{x}), \varphi(\boldsymbol{x}') \rangle$, where the explicit mapping $\varphi : \mathbb{R}^d \to \mathbb{R}^s$ is obtained by sampling from a distribution defined by the inverse Fourier transform of k. To scale kernel methods in the large sample case (e.g., $n \gg d$), the number of random features s is often taken to be larger than the original sample dimension d but much smaller than the sample size n to achieve computational efficiency in practice.[1] Accordingly, the random features model is a powerful tool for scaling up traditional kernel methods [10], [11], neural tangent kernel [12], [13], [14], graph neural networks [15], [16], and attention in Transformers [17], [18]. Interestingly, the random features model can be viewed as a class of two-layer neural networks with fixed weights in the first layer. This connection has important theoretical implications. It has been observed that deep neural networks (DNNs) exhibit certain intriguing phenomena such as the ability to fit random labels [19] and double descent [20] in the *over-parameterized* regime. Theoretical results [13], [21], [22], [23] for random features can be leveraged to explain these phenomena and provide an analysis of two-layer *over-parameterized* neural networks. Partly due to its far-reaching repercussions, the seminal work by Rahimi and Recht on RFF [9] won the Test-of-Time Award in the *Thirty-first Advances in Neural Information Processing Systems* (NeurIPS 2017).

F. Liu and J.A.K. Suykens are with the Department of Electrical Engineering (ESAT-STADIUS), KU Leuven, B-3001 Leuven, Belgium (email: {fanghui.liu;johan.suykens}@esat.kuleuven.be).

X. Huang is with Institute of Image Processing and Pattern Recognition, and also with Institute of Medical Robotics, Shanghai Jiao Tong University, Shanghai 200240, P.R. China (e-mail: xiaolinhuang@sjtu.edu.cn).

Y. Chen is with School of Operations Research and Information Engineering, Cornell University, Ithaca, NY 14850 USA (e-mail: yudong.chen@cornell.edu).

1. Random features model can be regarded as an over-parameterized model allowing for $s \gg n$, refer to Section 7 for details.

RFF spawns a new direction for kernel approximation, and the past ten years has witnessed a flurry of research papers devoted to this topic. On the algorithmic side, subsequent work has focused on improving the kernel approximation quality [24], [25] and decreasing the time and space complexities [26], [27]. Implementation of RFF has in fact been taken to the hardware level [28], [29]. On the theoretical side, a series of works aim to address the following two key questions:

1) **Approximation:** how many random features are needed to ensure high quality of kernel approximation?
2) **Generalization:** how many random features are needed to incur no loss in the expected risk of a learned estimator?

Here "no loss" means how large s should be for the (approximated) kernel estimator with s random features to be almost as good as the exact one. Much research effort has been devoted to this direction, including analyzing the kernel approximation error (the first question above) [9], [30], and studying the risk and generalization properties (the second question above) [11], [31]. Increasingly refined and general results have been obtained over the years. In the *Thirty-sixth International Conference on Machine Learning* (ICML 2019), Li et al. [31] were recognized by the Honorable Mentions (best paper finalist) for their unified theoretical analysis of RFF.

RFF has proved effective in a broad range of machine learning tasks. Given its remarkable empirical success and the rapid growth of the related literature, we believe it is desirable to have a comprehensive overview on this topic summarizing the progress in algorithm design and applications, and elucidating existing theoretical results and their underlying assumptions. With this goal in mind, in this survey we systematically review the work from the past ten years on the algorithms, theory and applications of random features methods. Figure 1 shows a schematic overview of the history of the work on random features in recent years. The main contributions of this survey include:

1) We provide an overview of a wide range of random features based algorithms, re-organize the formulation of representative approaches under a unifying framework for a direct understanding and comparison.
2) We summarize existing theoretical results on the kernel approximation error measured in various metrics, as well as results on generalization risk of kernel estimators. The underlying assumptions in these results are discussed in detail. In particular, we (partly) answer an open question in this topic: why good kernel approximation performance cannot lead to good generalization performance?
3) We systematically evaluate and compare the empirical performance of representative random features based algorithms under different experimental settings.
4) We discuss recent research trends on (high dimensional) random features in over-parameterized settings for understanding generalization properties of over-parameterized neural networks as well as the gaps in existing theoretical analysis. We view this topic as a promising research direction.

The remainder of this paper is organized as follows. Section 2 presents the preliminaries and a taxonomy of random features based algorithms. We review *data-independent* algorithms in Section 3 and *data-dependent* approaches in Section 4. In Section 5, we survey existing theoretical results on kernel approximation and generalization performance. Experimental comparisons of representative random features based methods are given in Section 6. In Section 7, we discuss recent results on random features in over-parameterized regimes. The paper is concluded in Section 8 with a discussion on future directions.

Table 1
Commonly used parameters and symbols.

Notation	Definition	Notation	Definition
n	number of samples	d	feature dimension
s	number of random features	λ	regularization parameter
k	(original) kernel function	$\tilde{k}$	(approximated) kernel function
$\boldsymbol{\omega}_i$	random feature	$\boldsymbol{\beta}_\lambda$	optimization variable
$\boldsymbol{x}$	data point	$\boldsymbol{y}$	label vector
ς	Gaussian kernel width	σ	activation function
$\boldsymbol{e}_i$	standard basis vector	u	$u := \langle \boldsymbol{x}, \boldsymbol{x}' \rangle / (\|\boldsymbol{x}\| \|\boldsymbol{x}'\|)$
$\boldsymbol{K}$	(original) kernel matrix	$\widetilde{\boldsymbol{K}}$	(approximated) kernel matrix
$\boldsymbol{\tau}$	$\boldsymbol{\tau} := \boldsymbol{x} - \boldsymbol{x}'$	τ	$\tau := \|\boldsymbol{\tau}\|_2$
$\boldsymbol{Z}$	random feature matrix	$\boldsymbol{W}$	transformation matrix
f_ρ	target function	ℓ	loss function
$f_{\boldsymbol{z},\lambda}$	(original) empirical functional	$\tilde{f}_{\boldsymbol{z},\lambda}$	(approximated) functional
$\mathcal{E}_{\boldsymbol{z}}$	empirical risk	$\mathcal{E}$	expected risk
$l_\lambda(\boldsymbol{\omega})$	ridge leverage function	$d^\lambda_{\boldsymbol{K}}$	effective dimension (matrix)
Σ	integral operator	$\mathcal{N}(\lambda)$	effective dimension (operator)
$\otimes$	tensor product	$\lesssim$	$\leq$ with a constant C times
α	convergence rate for λ	γ	rate for effective dimension

2 Preliminaries and Taxonomies

In this section, we introduce preliminaries on the problem setting and theoretical foundation of random features. We then present a taxonomy of existing random features based algorithms, which sets the stage for the subsequent discussion. A set of commonly used parameters is summarized in Table 1.

2.1 Problem Settings

Consider the following standard supervised learning setup. Let $\mathcal{X} \subset \mathbb{R}^d$ be a compact metric space of samples, and $\mathcal{Y} = \{-1, 1\}$ (in classification) or $\mathcal{Y} \subseteq \mathbb{R}$ (in regression) be the label space. We assume that a sample set $\{\boldsymbol{z}_i = (\boldsymbol{x}_i, y_i)\}_{i=1}^n$ is drawn from a non-degenerate unknown Borel probability measure ρ on $\mathcal{X} \times \mathcal{Y}$. Let $\mathcal{H}$ be a RKHS endowed with a positive definite kernel function $k(\cdot,\cdot)$, and $\boldsymbol{K} = [k(\boldsymbol{x}_i, \boldsymbol{x}_j)]_{i,j=1}^n$ be the kernel matrix associated with the samples. The *target function* of ρ is defined as $f_\rho(\boldsymbol{x}) = \int_{\mathcal{Y}} y \mathrm{d}\rho(y|\boldsymbol{x})$ for $\boldsymbol{x} \in \mathcal{X}$, where $\rho(\cdot|\boldsymbol{x})$ is the conditional distribution of y given $\boldsymbol{x}$. The typical empirical risk minimization problem is considered as

$$f_{\boldsymbol{z},\lambda} := \operatorname*{argmin}_{f \in \mathcal{H}} \left\{ \frac{1}{n} \sum_{i=1}^n \ell\big(y_i, f(\boldsymbol{x}_i)\big) + \lambda \|f\|_{\mathcal{H}}^2 \right\}, \qquad (1)$$

where $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ is a loss function and $\lambda \equiv \lambda(n) > 0$ is a regularization parameter. In learning theory, one typically assumes that $\lim_{n\to\infty} \lambda(n) = 0$ and adopts $\lambda := n^{-\alpha}$ with $\alpha \in (0, 1]$.

The loss function $\ell(y, f(\boldsymbol{x}))$ in Eq. (1) measures the quality of the prediction $f(\boldsymbol{x})$ at $\boldsymbol{x} \in \mathcal{X}$ with respect to the observed response y. Popular choices of ℓ include the squared loss $\ell(y, f(\boldsymbol{x})) = (y - f(\boldsymbol{x}))^2$ in kernel ridge regression (KRR) and the hinge loss $\ell(y, f(\boldsymbol{x})) = \max(0, 1 - yf(\boldsymbol{x}))$ in support vector machines (SVM), etc. For a given ℓ, the empirical risk functional on the sample set is defined as $\mathcal{E}_{\boldsymbol{z}}(f) = \frac{1}{n}\sum_{i=1}^n \ell(y_i, f(\boldsymbol{x}_i))$, and the corresponding expected risk is defined as $\mathcal{E}(f) = \int_{\mathcal{X}\times\mathcal{Y}} \ell(y, f(\boldsymbol{x})) \mathrm{d}\rho$. The statistical theory of supervised learning in an approximation theory view aims to understand the generalization property of $f_{\boldsymbol{z},\lambda}$ as an approximation

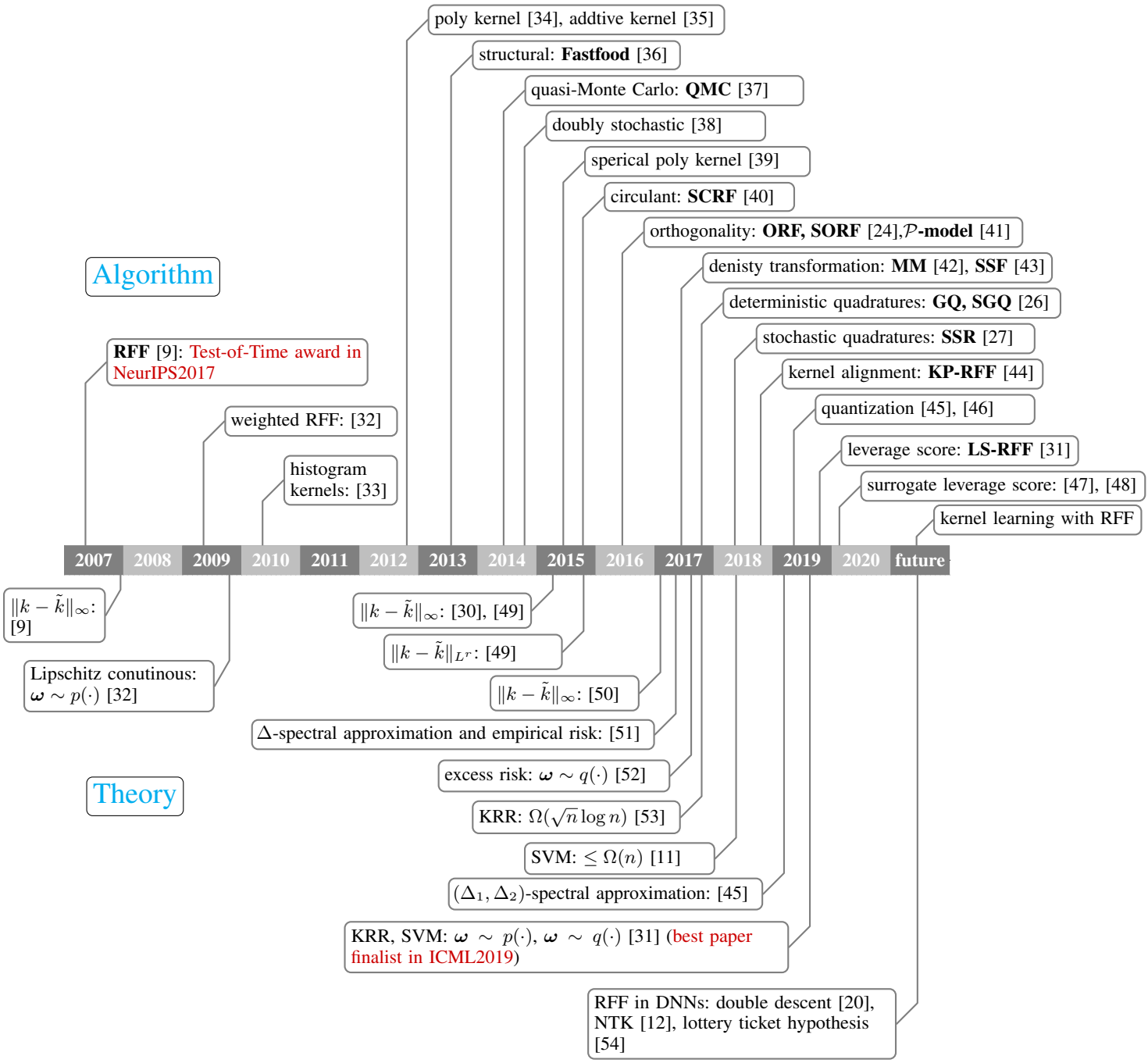


Figure 1. Timeline of representative work on the algorithms and theory of random features.

of the true target function f_ρ, which can be quantified by the excess risk $\mathcal{E}(f_{\boldsymbol{z},\lambda}) - \mathcal{E}(f_\rho)$, or the estimation error $\|f_{\boldsymbol{z},\lambda} - f_\rho\|^2$ in an appropriate norm $\|\cdot\|$.

Using an explicit randomized feature mapping $\varphi : \mathbb{R}^d \to \mathbb{R}^s$, one may approximate the kernel function $k(\boldsymbol{x}, \boldsymbol{x}')$ by $\tilde{k}(\boldsymbol{x}, \boldsymbol{x}') = \langle \varphi(\boldsymbol{x}), \varphi(\boldsymbol{x}') \rangle$. In this case, the approximate kernel $\tilde{k}(\cdot,\cdot)$ defines an RKHS $\widetilde{\mathcal{H}}$ (not necessarily contained in the RKHS $\mathcal{H}$ associated with the original kernel function k). With the above approximation, one solves the following approximate version of problem (1):

$$\tilde{f}_{\boldsymbol{z},\lambda} := \operatorname*{argmin}_{f \in \widetilde{\mathcal{H}}} \left\{ \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, f(\boldsymbol{x}_i)) + \lambda \|f\|_{\widetilde{\mathcal{H}}}^2 \right\}. \quad (2)$$

By the representer theorem [1], the above problem can be rewritten as a finite-dimensional empirical risk minimization problem

$$\boldsymbol{\beta}_\lambda := \operatorname*{argmin}_{\boldsymbol{\beta} \in \mathbb{R}^s} \frac{1}{n} \sum_{i=1}^{n} \ell\left(y_i, \boldsymbol{\beta}^\top \varphi(\boldsymbol{x}_i)\right) + \lambda \|\boldsymbol{\beta}\|_2^2. \quad (3)$$

For example, in least squares regression where ℓ is the squared loss, the first term in problem (3) is equivalent to $\|\boldsymbol{y} - \boldsymbol{Z}\boldsymbol{\beta}\|_2^2$, where $\boldsymbol{y} = [y_1, y_2, \cdots, y_n]^\top$ is the label vector and $\boldsymbol{Z} = [\varphi(\boldsymbol{x}_1), \cdots, \varphi(\boldsymbol{x}_n)]^\top \in \mathbb{R}^{n \times s}$ is the random feature matrix. This is a linear ridge regression problem in the space spanned by the random features, with the optimal prediction given by $\tilde{f}_{\boldsymbol{z},\lambda}(\boldsymbol{x}') = \boldsymbol{\beta}_\lambda^\top \varphi(\boldsymbol{x}')$ for a new data point $\boldsymbol{x}'$, where $\boldsymbol{\beta}_\lambda$ has the explicit expression $\boldsymbol{\beta}_\lambda = (\boldsymbol{Z}^\top \boldsymbol{Z} + n\lambda \boldsymbol{I})^{-1} \boldsymbol{Z}^\top \boldsymbol{y}$. For classification, one may take the sign to output the binary classification labels. Note

that problem (3) also corresponds to fixed-size kernel methods with feature map approximation (related to Nyström approximation) and estimation in the primal [2].

2.2 Theoretical Foundation of Random Features

The theoretical foundation of RFF builds on Bochner's celebrated characterization of positive definite functions.

Theorem 1 (Bochner's Theorem [55]). *A continuous and shift-invariant function $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is positive definite if and only if it can be represented as*

$$k(\boldsymbol{x} - \boldsymbol{x}') = \int_{\mathbb{R}^d} \exp\left(\mathrm{i}\boldsymbol{\omega}^\top(\boldsymbol{x} - \boldsymbol{x}')\right) \mu_k(\mathrm{d}\boldsymbol{\omega}) \,,$$

where μ_k is a positive finite measure on the frequencies $\boldsymbol{\omega}$.

According to Bochner's theorem, the spectral distribution μ_k of a stationary kernel k is the finite measure induced by a Fourier transform. By setting $k(0) = 1$, we may normalize μ_k to a probability density p (the Fourier transform associated with k), hence

$$\begin{aligned} k(\boldsymbol{x} - \boldsymbol{x}') &= \int_{\mathbb{R}^d} \exp\left(\mathrm{i}\boldsymbol{\omega}^\top(\boldsymbol{x} - \boldsymbol{x}')\right) \mu(\mathrm{d}\boldsymbol{\omega}) \\ &= \mathbb{E}_{\boldsymbol{\omega}\sim p(\cdot)}\left[\exp(\mathrm{i}\boldsymbol{\omega}^\top\boldsymbol{x}) \exp(\mathrm{i}\boldsymbol{\omega}^\top\boldsymbol{x}')^*\right] , \end{aligned} \tag{4}$$

where the symbol $\boldsymbol{z}^*$ denotes the complex conjugate of $\boldsymbol{z}$. The kernels used in practice are typically real-valued and thus the imaginary part in Eq. (4) can be discarded. According to Eq. (4), RFF makes use of the standard Monte Carlo sampling scheme to approximate $k(\boldsymbol{x}, \boldsymbol{x}')$. In particular, one uses the approximation

$$k(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{E}_{\boldsymbol{\omega}\sim p}[\varphi_p(\boldsymbol{x})^\top \varphi_p(\boldsymbol{x}')] \approx \tilde{k}_p(\boldsymbol{x}, \boldsymbol{x}') := \varphi_p(\boldsymbol{x})^\top \varphi_p(\boldsymbol{x}')$$

with the explicit feature mapping[2]

$$\varphi_p(\boldsymbol{x}) := \frac{1}{\sqrt{s}}\left[\exp(-\mathrm{i}\boldsymbol{\omega}_1^\top\boldsymbol{x}), \cdots, \exp(-\mathrm{i}\boldsymbol{\omega}_s^\top\boldsymbol{x})\right]^\top , \tag{5}$$

where $\{\boldsymbol{\omega}_i\}_{i=1}^s$ are sampled from $p(\cdot)$ *independently* of the training set. Consequently, the original kernel matrix $\boldsymbol{K} = [k(\boldsymbol{x}_i, \boldsymbol{x}_j)]_{n\times n}$ can be approximated by $\boldsymbol{K} \approx \widetilde{\boldsymbol{K}}_p = \boldsymbol{Z}_p\boldsymbol{Z}_p^\top$ with $\boldsymbol{Z}_p = [\varphi_p(\boldsymbol{x}_1), \cdots, \varphi_p(\boldsymbol{x}_n)]^\top \in \mathbb{R}^{n\times s}$. It is convenient to introduce the shorthand $z_p(\boldsymbol{\omega}_i, \boldsymbol{x}_j) := \exp(-\mathrm{i}\boldsymbol{\omega}_i^\top\boldsymbol{x}_j)$ such that $\varphi_p(\boldsymbol{x}) = 1/\sqrt{s}[z_p(\boldsymbol{\omega}_1, \boldsymbol{x}), \cdots, z_p(\boldsymbol{\omega}_s, \boldsymbol{x})]^\top$. With this notation, the approximate kernel $\tilde{k}_p(\boldsymbol{x}, \boldsymbol{x}')$ can be rewritten as $\tilde{k}_p(\boldsymbol{x}, \boldsymbol{x}') = \frac{1}{s}\sum_{i=1}^s z_p(\boldsymbol{\omega}_i, \boldsymbol{x}) z_p(\boldsymbol{\omega}_i, \boldsymbol{x}')$.

A similar characterization in Eq. (4) is available for rotation-invariant kernels, where the Fourier basis functions are *spherical harmonics* [56], [57]. Here rotation-invariant kernels are dot-product kernels defined on the unit sphere $\mathcal{X} = \mathbb{S}^{d-1} := \{\boldsymbol{x} \in \mathbb{R}^d : \|\boldsymbol{x}\|_2 = 1\}$, and can be represented as a non-negative expansion with spherical harmonics, refer to the book [58] for details.

Theorem 2 ([56]). *A rotation-invariant continuous function $k : \mathbb{S}^{d-1} \times \mathbb{S}^{d-1} \to \mathbb{R}$ is positive definite if and only if it has a symmetric non-negative expansion into spherical harmonics $Y_{\ell,m}^d$, that is*

$$k(\boldsymbol{x}, \boldsymbol{x}') \equiv k(\langle \boldsymbol{x}, \boldsymbol{x}'\rangle) = \sum_{i=0}^{\infty} \Lambda_i \sum_{j=1}^{N(d,i)} Y_{i,j}(\boldsymbol{x}) Y_{i,j}(\boldsymbol{x}') \,,$$

where $\Lambda_i \geq 0$ are the Fourier coefficients, $Y_{i,j}$ is the spherical harmonics, and $N(d,i) = \frac{2i+d-2}{i}\begin{pmatrix} i+d-3 \\ d-2 \end{pmatrix}$.

Note that, dot product kernels defined in $\mathbb{R}^d$ do not belong to the *rotation-invariant* class. Nevertheless, by virtue of the neural network structure under Gaussian initialization, some dot product kernels defined on $\mathbb{R}^d$ are able to benefit from the sampling framework behind RFF. Given a two-layer network of the form $f(\boldsymbol{x}; \boldsymbol{\theta}) = \sqrt{\frac{2}{s}}\sum_{j=1}^s a_j \sigma(\boldsymbol{\omega}_j^\top\boldsymbol{x})$ with s neurons (notation chosen to be consistent with the number of random features), for some activation function σ and $\boldsymbol{x} \in \mathbb{R}^d$, when $\boldsymbol{\omega} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_d)$ are fixed and only the second layer (parameters $\boldsymbol{a}$) are optimized[3], this actually corresponds to random features approximation

$$k(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{E}_{\boldsymbol{\omega}\sim\mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_d)}[\sigma(\boldsymbol{\omega}^\top\boldsymbol{x})\sigma(\boldsymbol{\omega}^\top\boldsymbol{x}')] \,, \tag{6}$$

where the nonlinear activation function $\sigma(\cdot)$ depends on the kernel type such that $\varphi(\boldsymbol{x}_i) := \sigma(\boldsymbol{W}\boldsymbol{x}_i)$ in Eq. (5), by denoting the transformation matrix $\boldsymbol{W} := [\boldsymbol{\omega}_1, \boldsymbol{\omega}_2, \cdots, \boldsymbol{\omega}_s]^\top \in \mathbb{R}^{s\times d}$. The formulation in (6) is quite general to cover a series of kernels by various activation functions. For example, if we take $\sigma(x) = [\cos(x), \sin(x)]^\top$, Eq. (6) corresponds to the Gaussian kernel, which is the standard RFF model [9] for Gaussian kernel approximation. If we consider the commonly used ReLU activation $\sigma(x) = \max\{0, x\}$ in neural networks, Eq. (6) corresponds to the first order arc-cosine kernel, termed as $k(\boldsymbol{x}, \boldsymbol{x}') \equiv \kappa_1(u) = \frac{1}{\pi}(u(\pi - \arccos(u)) + \sqrt{1-u^2})$ by setting $u := \langle \boldsymbol{x}, \boldsymbol{x}'\rangle/(\|\boldsymbol{x}\|\|\boldsymbol{x}'\|)$. If the Heaviside step function $\sigma(x) = \frac{1}{2}(1 + \mathrm{sign}(x))$ is used, Eq. (6) corresponds to the zeroth order arc-cosine kernel, termed as $k(\boldsymbol{x}, \boldsymbol{x}') \equiv \kappa_0(u) = 1 - \frac{1}{\pi}\arccos(u)$ by setting $u := \langle \boldsymbol{x}, \boldsymbol{x}'\rangle/(\|\boldsymbol{x}\|\|\boldsymbol{x}'\|)$, refer to arc-cosine kernels [60] for details. If we take other activation functions used in neural networks, e.g., erf activations [61], GELU [62] in Eq. (6), such two-layer neural network also corresponds to a kernel. In this case, the standard RFF model is still valid (via Monte Carlo sampling from a Gaussian distribution) for these non-stationary kernels.

Further, for a fully-connected deep neural network (more than two layers) and fixed random weights before the output layer, if the hidden layers are wide enough, one can still approach a kernel obtained by letting the widths tend to infinity [63], [64]. If both intermediate layers and the output layer are trained by (stochastic) gradient descent, for the network $f(\boldsymbol{x}; \boldsymbol{\theta})$ with large enough s, the model remains close to its linearization around its random initialization throughout training, known as *lazy training* regime [65]. Learning is then equivalent to a kernel method with another architecture-specific kernel, known as *neural tangent kernel* (NTK, [12]). Interestingly, NTK for two-layer ReLU networks [66] can be constructed by arc-cosine kernels, i.e., $k(\boldsymbol{x}, \boldsymbol{x}') = \|\boldsymbol{x}\|\|\boldsymbol{x}'\|[u\kappa_0(u) + \kappa_1(u)]$. In fact, there is an interesting line of work showing insightful connections between kernel methods and (over-parameterized) neural networks, but this is out of scope of this survey on random features. We suggest the readers refer to some recent literature [13], [67], [68] for details.

Further, if we consider the general non-stationary kernels [69], [70], the spectral representation can be generalized by introducing two random variables $\boldsymbol{\omega}$ and $\boldsymbol{\omega}'$.

2. The subscript in φ_p, $\boldsymbol{Z}_p$, k_p (and other symbols) emphasizes the dependence on the distribution $p(\cdot)$ but can be omitted for notational simplicity.

3. Extreme learning machine [59] is another structure in a two-layer feedforward neural network by randomly hidden nodes.

Theorem 3. *([70], [71], [72]) A non-stationary kernel k is positive definite if and only if it admits*

$$k(\boldsymbol{x}, \boldsymbol{x}') = \int_{\mathbb{R}^d \times \mathbb{R}^d} \exp\left(\mathrm{i}\left(\boldsymbol{\omega}^\top \boldsymbol{x} - \boldsymbol{\omega}'^\top \boldsymbol{x}'\right)\right) \mu_{\Psi_k}(\mathrm{d}\boldsymbol{\omega}, \mathrm{d}\boldsymbol{\omega}'),$$

where μ_{Ψ_k} is the Lebesgue-Stieltjes measure on the product space $\mathbb{R}^d \times \mathbb{R}^d$ associated to some positive definite function $\Psi_k(\boldsymbol{\omega}, \boldsymbol{\omega}')$ with bounded variations.

2.3 Commonly used kernels in Random Features

Random features based algorithms often consider the following kernels:

i) Gaussian kernel: Arguably the most important member of shift-invariant kernels, the Gaussian kernel is given by

$$k(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|_2^2}{2\varsigma^2}\right),$$

where $\varsigma > 0$ is the kernel width. The density (see Theorem 1 or Eq. (6)) associated with the Gaussian kernel is Gaussian $\boldsymbol{\omega} \sim \mathcal{N}(0, \varsigma^{-2}\boldsymbol{I}_d)$.

ii) arc-cosine kernels: This class admits Eq. (6) by sampling from the Gaussian distribution $\mathcal{N}(0, \boldsymbol{I}_d)$, that can be connected to a two-layer neural networks with various activation functions. Following [60], we define the b-order arc-cosine kernel by

$$k(\boldsymbol{x}, \boldsymbol{x}') = \frac{1}{\pi}\|\boldsymbol{x}\|_2^b \|\boldsymbol{x}'\|_2^b J_b(\theta),$$

where $\theta = \cos^{-1}\left(\frac{\boldsymbol{x}^\top \boldsymbol{x}'}{\|\boldsymbol{x}\|_2 \|\boldsymbol{x}'\|_2}\right)$ and

$$J_b(\theta) = (-1)^b (\sin\theta)^{2b+1} \left(\frac{1}{\sin\theta}\frac{\partial}{\partial\theta}\right)^b \left(\frac{\pi - \theta}{\sin\theta}\right).$$

Most common in practice are the zeroth order ($b = 0$) and first order ($b = 1$) arc-cosine kernels. The zeroth order kernel is given explicitly by

$$k(\boldsymbol{x}, \boldsymbol{x}') = 1 - \frac{\theta}{\pi},$$

and the first order kernel is

$$k(\boldsymbol{x}, \boldsymbol{x}') = \frac{1}{\pi}\|\boldsymbol{x}\|_2 \|\boldsymbol{x}'\|_2 \left(\sin\theta + (\pi - \theta)\cos\theta\right).$$

iii) Polynomial kernel: This is a widely used family of non-stationary kernels given by

$$k(\boldsymbol{x}, \boldsymbol{x}') = (1 + \langle \boldsymbol{x}, \boldsymbol{x}' \rangle)^b,$$

where b is the order of the polynomial.

Note that, dot-product kernel defined in $\mathbb{R}^d$ admit neither *spherical harmonics* nor Eq. (6). As a result, random features for polynomial kernels work in different theoretical foundations and settings, and have been studied in a smaller number of papers, including Maclaurin expansion [34], the tensor sketch technique [73], [74], and oblivious subspace embedding [75], [76]. Interestingly, if the data are ℓ_2 normalized, dot product kernels defined in $\mathbb{R}^d$ can be transformed as stationary but indefinite (real, symmetric, but not positive definite) on the unit sphere[4]. The related random features based algorithms under this setting provide biased estimators [39], [77], or unbiased estimation [78].

2.4 Taxonomy of random features based algorithms

The key step in random features based algorithms is constructing the following random feature mapping

$$\varphi(\boldsymbol{x}) := \frac{1}{\sqrt{s}}\left[a_1 \exp(-\mathrm{i}\boldsymbol{\omega}_1^\top \boldsymbol{x}), \cdots, a_s \exp(-\mathrm{i}\boldsymbol{\omega}_s^\top \boldsymbol{x})\right]^\top \quad (7)$$

so as to approximate the integral (4). Random features $\{\boldsymbol{\omega}_i\}_{i=1}^s$ can be formulated as the feature matrix $\boldsymbol{W} = [\boldsymbol{\omega}_1, \cdots, \boldsymbol{\omega}_s]^\top \in \mathbb{R}^{s \times d}$ in a compact form. Existing algorithms differ in how they select the points $\boldsymbol{\omega}_i$ (the transformation matrix $\boldsymbol{W}$) and weights a_i. Figure 2 presents a taxonomy of some representative random features based algorithms. They can be grouped into two categories, *data-independent* algorithms and *data-dependent* algorithms, based on whether or not the selection of $\boldsymbol{\omega}_i$ and a_i is independent of the training data.

Data-independent random features based algorithms can be further categorized into three classes according to their sampling strategy:

i) *Monte Carlo sampling*: The points $\{\boldsymbol{\omega}_i\}_{i=1}^s$ are sampled from $p(\cdot)$ in Eq. (4) (see the red box in Figure 2). In particular, to approximate the Gaussian kernel by RFF [9], these points are sampled from the Gaussian distribution $p = \mathcal{N}(0, \varsigma^{-2}\boldsymbol{I}_d)$, with the weights being equal, i.e., $a_i \equiv 1$ in Eq. (7). To reduce the storage and time complexity, one may replace the dense Gaussian matrix in RFF by structural matrices; see, e.g., Fastfood [36] using Hadamard matrices as well as its general version $\mathcal{P}$-model [41]. An alternative approach is using circulant matrices; see, e.g., Signed Circulant Random Features (SCRF) [40]. To improve the approximation quality, a simple and effective approach is to use an ℓ_2-normalization scheme, which leads to Normalized RFF (NRFF) [79]. Another powerful technique for variance reduction is orthogonalization to decrease the randomness in Monte Carlo sampling. Typical algorithms include Orthogonal Random Features (ORF) [24] by employing an orthogonality constraint to the random Gaussian matrix, Structural ORF (SORF) [24], [91], and Random Orthogonal Embeddings (ROM) [80].

ii) *Quasi-Monte Carlo sampling*: This is a typical sampling scheme in sampling theory [92] to reduce the randomness in Monte Carlo sampling for variance reduction. It can significantly improve the convergence of Monte Carlo sampling by virtue of a low-discrepancy sequence $\boldsymbol{t}_1, \boldsymbol{t}_2, \cdots, \boldsymbol{t}_s \in [0, 1]^d$ instead of a uniform sampling sequence over the unit cube to construct the sample points; see the integral representation in the green box in Figure 2. Based on this representation, it can be used for kernel approximation, as conducted by [25]. Subsequently, Lyu [43] proposes Spherical Structural Features (SSF), which generates asymptotically uniformly distributed points on $\mathbb{S}^{d-1}$ to achieve better convergence rate and approximation quality. The Moment Matching (MM) scheme [42] is based on the same integral representation but uses a d-dimensional refined uniform sampling sequence $\{\boldsymbol{t}_i\}_{i=1}^s$ instead of a low discrepancy sequence. Strictly speaking, SSF and MM go beyond the QMC framework. Nevertheless, these methods share the same integration formulation with QMC over the unit cube and thus we include them here for a streamlined presentation.

iii) *Quadrature based methods*: Numerical integration techniques can be also used to approximate the integral representation in Eq. (4). These techniques may involve *deterministic* selection of

4. This setting cannot ensure the data are i.i.d on the unit sphere, which is different from the setting of previously discussed rotation invariant kernels.

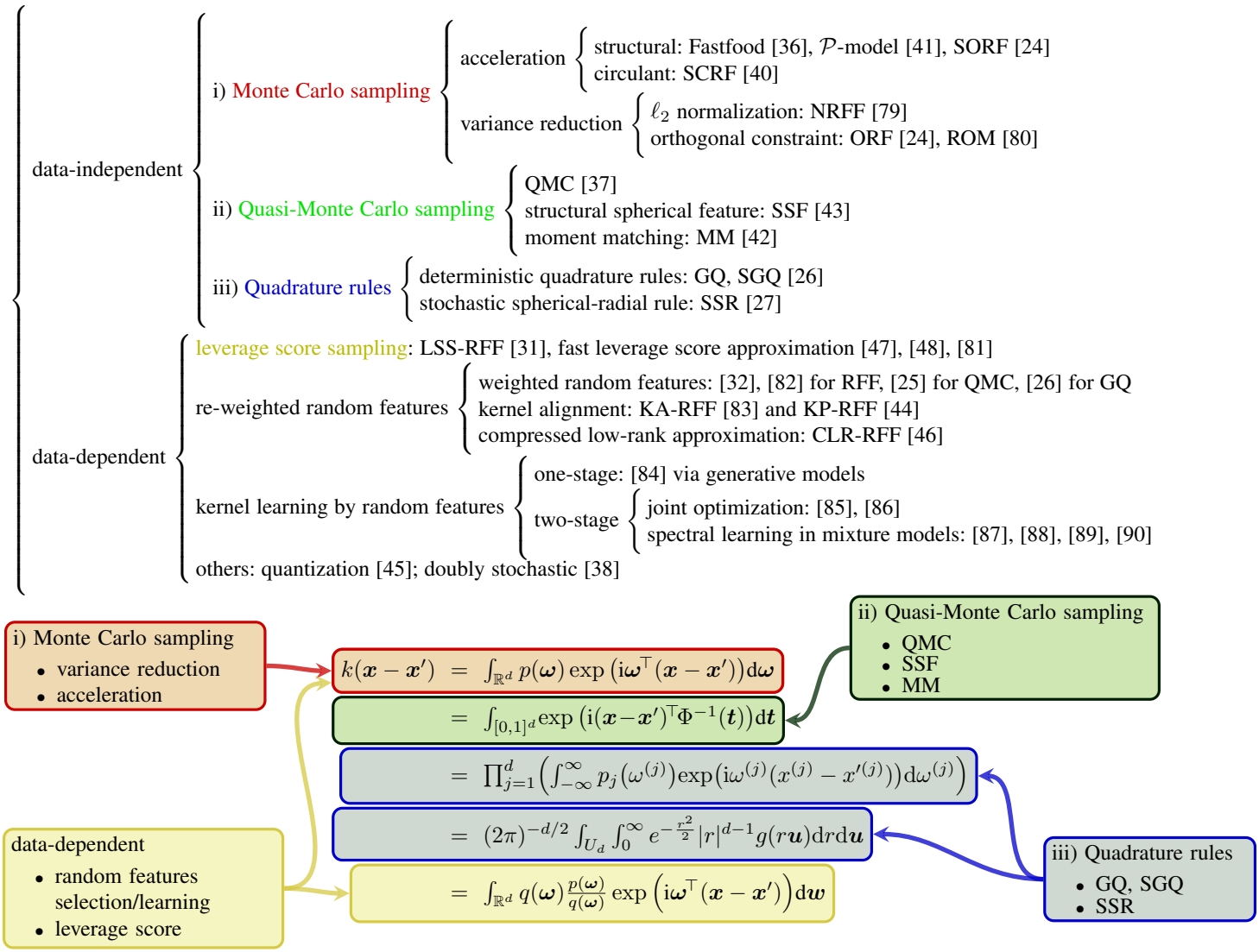


Figure 2. A taxonomy of representative random features based algorithms.

the points and weights, e.g., by using Gaussian Quadrature (GQ) [26] or Sparse Grids Quadrature (SGQ) [26] over each dimension (their integration formulation can be found in the first blue box in Figure 2). The selection can also be *randomized*. For example, in the work [27], the d-dimensional integration in Eq. (4) is transformed to a double integral, and then approximated by using the Stochastic Spherical-Radial (SSR) rule (see the second blue box in Figure 2).

Data-dependent algorithms use the training data to guide the selection of points and weights in the random features for better approximation quality and/or generalization performance. These algorithms can be grouped into three classes according to how the random features are generated.

i) Leverage score sampling: Built upon the importance sampling framework, this class of algorithm replaces the original distribution $p(\boldsymbol{\omega})$ by a carefully chosen distribution $q(\boldsymbol{\omega})$ constructed using leverage scores [51], [52] (see the yellow box in Figure 2). The representative approach in this class is Leverage Score based RFF (LS-RFF) [31], and its accelerated version [47], [81].

ii) Re-weighted random feature selection: Here the basic idea is to re-weight the random features by solving a constrained optimization problem. Examples of this approach include weighted RFF [32], [82], weighted QMC [25], and weighted GQ [26]. Note that these algorithms directly learn the weights of pre-given random features. Another line of methods re-weight the random features using a two-step procedure: i) "up-projection": first generate a large set of random features $\{\boldsymbol{\omega}_i\}_{i=1}^J$; ii) "compression": then reduce these features to a small number (e.g., $10^2 \sim 10^3$) in a data-dependent manner, e.g., by using kernel alignment [83], kernel polarization [44], or compressed low-rank approximation [46].

iii) Kernel learning by random features: This class of methods aim to learn the spectral distribution of kernel *from the data* so as to achieve better similarity representation and prediction. Note that these methods learn both the weights and the distribution of the features, and hence differ from the other random features selection methods mentioned above, which assume that the candidate features are generated from a pre-given distribution and only learn the weights of these features. Representative approaches for kernel learning involve a *one-stage* [84] or *two-stage* procedure [85], [86], [87], [88], [89], [90]. From a more general point of view, the aforementioned *re-weighted random features selection* methods can also be classified into this class. Since these methods belong to the broad area of kernel learning instead of kernel approximation,

we do not detail them in this survey.

Besides the above three main categories, other data-dependent approaches include the following. i) Quantization random features [45]: Given a memory budget, this method quantizes RFF for Gaussian kernel approximation. A key observation from this work is that random features achieve better generalization performance than Nyström approximation [93] under the same memory space. ii) Doubly stochastic random features [38]: This method uses two sources of stochasticity, one from sampling data points by stochastic gradient descent (SGD), and the other from using RFF to approximate the kernel. This scheme has been used for Kernel PCA approximation [94], and can be further extended to triply stochastic scheme for multiple kernel approximation [95].

3 Data-independent Algorithms

In this section, we discuss data-independent algorithms in a unified framework based on the transformation matrix $\boldsymbol{W}$, that plays an important role in constructing the mapping $\varphi(\cdot)$ in Eq. (7) and determining how well the estimated kernel converges to the actual kernel. Table 2 reports various random features based algorithms in terms of the class of kernels they apply to (in theory) as well as their space and time complexities for computing the feature mapping $\boldsymbol{W}\boldsymbol{x}$ for a given $\boldsymbol{x} \in \mathcal{X}$. In Table 2, we also summarize the *variance reduction* properties of these algorithms, i.e., whether the variance of the resulting kernel estimator is smaller than the standard RFF. Before proceeding, we introduce some notations and definitions. When discussing a stationary kernel function $k(\boldsymbol{x}, \boldsymbol{x}') = k(\boldsymbol{x} - \boldsymbol{x}')$, we use the convenient shorthands $\boldsymbol{\tau} := \boldsymbol{x} - \boldsymbol{x}'$ and $\tau := \|\boldsymbol{\tau}\|_2$. For a random features algorithm A with frequencies $\{\boldsymbol{\omega}_i\}_{i=1}^s$ sampled from a distribution $\mu(\cdot)$, we define its expectation $\mathbb{E}(A) := \mathbb{E}[k(\boldsymbol{\tau})] = \mathbb{E}_{\boldsymbol{\omega}\sim\mu}\left[1/s \sum_{i=1}^s \cos(\boldsymbol{\omega}_i^\top \boldsymbol{\tau})\right]$ and variance $\mathbb{V}[A] := \mathbb{V}[k(\boldsymbol{\tau})] = \mathbb{V}\left[\frac{1}{s}\sum_{i=1}^s \cos(\boldsymbol{\omega}^\top \boldsymbol{\tau})\right]$.

3.1 Monte Carlo sampling based approaches

We describe several representative data-independent algorithms based on Monte Carlo sampling, using the Gaussian kernel $k(\boldsymbol{x}, \boldsymbol{x}') = k(\boldsymbol{\tau}) = \exp(-\|\boldsymbol{\tau}\|_2^2/2\varsigma^2)$ as an example. Note that these algorithms often apply to more general classes of kernels, as summarized in Table 2.

RFF [9]: For Gaussian kernels, RFF directly samples the random features from a Gaussian distribution (corresponds to the inverse Fourier transform): $\{\boldsymbol{\omega}\}_{i=1}^s \sim p(\boldsymbol{\omega})$. In particular, the corresponding transformation matrix is given by

$$\boldsymbol{W}_{\text{RFF}} = \frac{1}{\varsigma}\boldsymbol{G}\,, \tag{8}$$

where $\boldsymbol{G} \in \mathbb{R}^{s\times d}$ is a (dense) Gaussian matrix with $G_{ij} \sim \mathcal{N}(0,1)$. For other stationary kernels, the associated $p(\cdot)$ corresponds to the specific distribution given by the Bochner's Theorem. For example, the Laplacian kernel $k(\boldsymbol{\tau}) = \exp(-\|\boldsymbol{\tau}\|_1/\varsigma)$ is associated with a Cauchy distribution. RFF is unbiased, i.e., $\mathbb{E}[\text{RFF}] = \exp(-\|\boldsymbol{\tau}\|_2^2/2\varsigma^2)$, and the corresponding variance is $\mathbb{V}[\text{RFF}] = (1 - e^{-\tau^2})^2/2s$.

Fastfood [36]: By observing the similarity between the dense Gaussian matrix and Hadamard matrices with diagonal Gaussian matrices, Le et al. [36] firstly introduce Hadamard and diagonal matrices to speed up the construction of dense Gaussian matrices in RFF, especially in high dimensions (e.g., $d \geq 1000$). In particular, $\boldsymbol{W}$ used in Eq. (8) is substituted by

$$\boldsymbol{W}_{\text{Fastfood}} = \frac{1}{\varsigma}\boldsymbol{B}_1\boldsymbol{H}\boldsymbol{G}\boldsymbol{\Gamma}\boldsymbol{H}\boldsymbol{B}_2\,, \tag{9}$$

where $\boldsymbol{H}$ is the Walsh-Hadamard matrix admitting fast multiplication in $\mathcal{O}(d\log d)$ time, and $\boldsymbol{\Gamma} \in \{0,1\}^{d\times d}$ is a permutation matrix that decorrelates the eigen-systems of two Hadamard matrices. The three *diagonal* random matrices $\boldsymbol{G}$, $\boldsymbol{B}_1$ and $\boldsymbol{B}_2$ are specified as follows: $\boldsymbol{G}$ has independent Gaussian entries drawn from $\mathcal{N}(0,1)$; $\boldsymbol{B}_1$ is a random scaling matrix with $(\boldsymbol{B}_1)_{ii} = \|\boldsymbol{\omega}_i\|_2/\|\boldsymbol{G}\|_{\text{F}}$, which encodes the spectral properties of the associated kernel; $\boldsymbol{B}_2$ is a binary decorrelation matrix with independent random $\{\pm 1\}$ entries. FastFood is an unbiased estimator, but may have a larger variance than RFF:

$$\mathbb{V}[\text{Fastfood}] - \mathbb{V}[\text{RFF}] \leq \frac{6\tau^4}{s}\left(e^{-\tau^2} + \frac{\tau^2}{3}\right)\,,$$

which converges at an $\mathcal{O}(1/s)$ rate.

$\mathcal{P}$-model [41]: A general version of Fastfood, the $\mathcal{P}$-model constructs the transformation matrix as

$$\boldsymbol{W}_{\mathcal{P}} = [\boldsymbol{g}^\top\boldsymbol{P}_1, \boldsymbol{g}^\top\boldsymbol{P}_2, \cdots, \boldsymbol{g}^\top\boldsymbol{P}_s]^\top \in \mathbb{R}^{s\times d}\,,$$

where $\boldsymbol{g}$ is a Gaussian random vector of length a and $\mathcal{P} = \{\boldsymbol{P}_i\}_{i=1}^s$ is a sequence of a-by-d matrices each with unit ℓ_2 norm columns. Fastfood can viewed as a special case of the $\mathcal{P}$-model: the matrix $\boldsymbol{H}\boldsymbol{G}$ in Eq. (9) can be constructed by using a fixed budget of randomness in $\boldsymbol{g}$ and letting each $\boldsymbol{P}_i$ be a random diagonal matrix with diagonal entries of the form $H_{i1}, H_{i2}, \ldots, H_{id}$. The $\mathcal{P}$-model is unbiased and its variance is close to that of RFF with an $\mathcal{O}(1/d)$ convergence rate

$$\left|\mathbb{V}[\mathcal{P}\text{-model}] - \mathbb{V}[\text{RFF}]\right| = \mathcal{O}\left(1/d\right)\,.$$

SCRF [40]: It accelerates the construction of random features by using circulant matrices. The transformation matrix is

$$\boldsymbol{W}_{\text{SCRF}} = [\boldsymbol{\nu}\otimes\mathcal{C}(\boldsymbol{\omega}_1), \boldsymbol{\nu}\otimes\mathcal{C}(\boldsymbol{\omega}_2), \cdots, \boldsymbol{\nu}\otimes\mathcal{C}(\boldsymbol{\omega}_t)]^\top \in \mathbb{R}^{td\times d}\,,$$

where $\otimes$ denotes the tensor product, $\boldsymbol{\nu} = [\nu_1, \nu_2, \ldots, \nu_d]$ is a Rademacher vector with $\mathbb{P}(\nu_i = \pm 1) = 1/2$, and $\mathcal{C}(\boldsymbol{w}_i) \in \mathbb{R}^{d\times d}$ is a circulant matrix generated by the vector $\boldsymbol{\omega}_i \sim \mathcal{N}(0, \varsigma^{-2}\boldsymbol{I}_d)$. Thanks to the circulant structure, we only need $\mathcal{O}(s)$ space to store the feature mapping matrix $\boldsymbol{W}_{\text{SCRF}}$ with $s = td$. Note that $\mathcal{C}(\boldsymbol{w}_i)$ can be diagonalized using the Discrete Fourier Transform for $\boldsymbol{\omega}_i$. SCRF is unbiased and has the same variance as RFF.

The above three approaches are designed to accelerate the computation of RFF. We next overview representative methods that aim for better approximation performance than RFF.

NRFF [79]: It normalizes the input data to have unit ℓ_2 norm before constructing the random Fourier features. With normalized data, the Gaussian kernel can be computed as

$$k(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\frac{1}{\varsigma^2}\left(1 - \frac{\boldsymbol{x}^\top\boldsymbol{x}'}{\|\boldsymbol{x}\|_2\|\boldsymbol{x}'\|_2}\right)\right)\,,$$

which is related to the normalized linear kernel [39], [79]. Albeit simple, NRFF is effective in variance reduction and in particular satisfies

$$\mathbb{V}[\text{NRFF}] = \mathbb{V}[\text{RFF}] - \frac{1}{4s}e^{-\tau^2}(3 - e^{-2\tau^2})\,.$$

Table 2
Comparison of different kernel approximation methods on space and time complexities to obtain $\boldsymbol{W}\boldsymbol{x}$.

Method	Kernels (in theory)	Extra Memory	Time	Lower variance than RFF
Random Fourier Features (RFF) [9]	shift-invariant kernels	$\mathcal{O}(sd)$	$\mathcal{O}(sd)$	-
Quasi-Monte Carlo (QMC) [37]	shift-invariant kernels	$\mathcal{O}(sd)$	$\mathcal{O}(sd)$	Yes
Normalized RFF (NRFF) [79]	Gaussian kernel	$\mathcal{O}(sd)$	$\mathcal{O}(sd)$	Yes
Moment matching (MM) [42]	shift-invariant kernels	$\mathcal{O}(sd)$	$\mathcal{O}(sd)$	Yes
Orthogonal Random Feature (ORF) [24]	Gaussian kernel	$\mathcal{O}(sd)$	$\mathcal{O}(sd)$	Yes
Fastfood [36]	Gaussian kernel	$\mathcal{O}(s)$	$\mathcal{O}(s\log d)$	No
Spherical Structured Features (SSF) [43]	shift and rotation-invariant kernels	$\mathcal{O}(s)$	$\mathcal{O}(s\log d)$	Yes
Structured ORF (SORF) [24], [91]	shift and rotation-invariant kernels	$\mathcal{O}(s)$	$\mathcal{O}(s\log d)$	Unknown
Signed Circulant (SCRF) [40]	shift-invariant kernels	$\mathcal{O}(s)$	$\mathcal{O}(s\log d)$	The same
$\mathcal{P}$-model [41]	shift and rotation-invariant kernels	$\mathcal{O}(s)$	$\mathcal{O}(s\log d)$	No
Random Orthogonal Embeddings (ROM) [80]	rotation-invariant kernels	$\mathcal{O}(d)$	$\mathcal{O}(d\log d)$	Yes
Gaussian Quadrature (GQ), Sparse Grids Quadrature (SGQ) [26]	shift invariant kernels	$\mathcal{O}(d)$	$\mathcal{O}(d\log d)$	Yes
Stochastic Spherical-Radial rules (SSR) [27]	shift and rotation-invariant kernels	$\mathcal{O}(d)$	$\mathcal{O}(d\log d)$	Yes

ORF [24]: It imposes orthogonality on random features for the Gaussian kernel and has the transformation matrix

$$\boldsymbol{W}_{\text{ORF}} = \frac{1}{\varsigma}\boldsymbol{S}\boldsymbol{Q}\,,$$

where $\boldsymbol{Q}$ is a uniformly distributed random orthogonal matrix, and $\boldsymbol{S}$ is a diagonal matrix with diagonal entries sampled *i.i.d* from the χ-distribution with d degrees of freedom. This orthogonality constraint is useful in reducing the approximation error in random features. It is also considered in [96] for unifying orthogonal Monte Carlo methods. ORF is unbiased and with variance bounded by

$$\mathbb{V}[\text{ORF}] - \mathbb{V}[\text{RFF}] \leq \frac{1}{s}\left(\frac{g(\tau)}{d} - \frac{(d-1)e^{-\tau^2}\tau^4}{2d}\right),$$

where we have $g(\tau) = e^{\tau^2}\left(\tau^8+6\tau^6+7\tau^4+\tau^2\right)/4 + e^{\tau^2}\tau^4\left(\tau^6+2\tau^4\right)/2d$. It can be seen that the variance reduction property $\text{Var}[\text{ORF}] < \text{Var}[\text{RFF}]$ holds under some conditions, e.g., when d is large and τ is small. For a large d, the ratio of the variances of ORF and RFF can be approximated by

$$\frac{\mathbb{V}[\text{ORF}]}{\mathbb{V}[\text{RFF}]} \approx 1 - \frac{(s-1)e^{-\tau^2}\tau^4}{d\left(1-e^{-\tau^2}\right)^2}\,. \tag{10}$$

Choromanski et al. [97] further improve the variance bound to

$$\begin{aligned}\mathbb{V}[\text{RFF}]-\mathbb{V}[\text{ORF}] = \\ \frac{s-1}{s}\mathbb{E}_{R_1,R_2}\left[\frac{J_{\frac{d}{2}-1}(\sqrt{R_1^2+R_2^2}\tau)\Gamma(d/2)}{(\sqrt{R_1^2+R_2^2}\tau/2)^{\frac{d}{2}-1}}\right] \\ - \frac{s-1}{s}\mathbb{E}_{R_1}\left[\frac{J_{\frac{d}{2}-1}(R_1\tau)\,\Gamma(d/2)}{(R_1\tau/2)^{\frac{d}{2}-1}}\right]^2,\end{aligned} \tag{11}$$

where J_d is the Bessel function of the first kind of degree d, and R_1 and R_2 are two independent scalar random variables satisfying $\boldsymbol{\omega}_1 = R_1\boldsymbol{v}$ and $\boldsymbol{\omega}_2 = R_2\boldsymbol{v}$ with $\boldsymbol{\omega}_1, \boldsymbol{\omega}_2 \sim \mathcal{N}(0, \varsigma^{-2}\boldsymbol{I}_d)$ and $\boldsymbol{v} \sim \text{Unif}(\mathcal{S}^{d-1})$. According to Eq. (11), the property $\mathbb{V}[\text{ORF}] < \mathbb{V}[\text{RFF}]$ holds asymptotically in cases: i) a fixed d and a small enough τ with $\mathbb{E}[\|\boldsymbol{\omega}\|_2^4] \leq \infty$; ii) a fixed $\tau < \frac{1}{4\sqrt{c}}$ with some constant c and a large d, in which case we have

$$\mathbb{V}[\text{RFF}] - \mathbb{V}[\text{ORF}] = \frac{s-1}{s}\left(\frac{1}{2d}\frac{\tau^4}{\varsigma^2}e^{-\frac{\tau^2}{\varsigma^2}} + \mathcal{O}\Big(\frac{1}{d}\Big)\right).$$

SORF [24], [91]: It replaces the random orthogonal matrices used in ORF by a class of structured matrices akin to those in Fastfood. The transformation matrix of SORF is given by

$$\boldsymbol{W}_{\text{SORF}} = \frac{\sqrt{d}}{\varsigma}\boldsymbol{H}\boldsymbol{D}_1\boldsymbol{H}\boldsymbol{D}_2\boldsymbol{H}\boldsymbol{D}_3\,, \tag{12}$$

where $\boldsymbol{H}$ is the normalized Walsh-Hadamard matrix and $\boldsymbol{D}_i \in \mathbb{R}^{d\times d}$, $i = 1,2,3$ are diagonal "sign-flipping" matrices, of which each diagonal entry is sampled from the Rademacher distribution. Bojarski et al. [91] consider more general structures for the three blocks of matrices $\boldsymbol{H}\boldsymbol{D}_i$ in Eq. (12). Note that each block plays a different role. The first block $\boldsymbol{H}\boldsymbol{D}_1$ satisfies $\Pr\left[\|\boldsymbol{H}\boldsymbol{D}_1\boldsymbol{x}\|_\infty > \frac{\log d}{\sqrt{d}}\right] \leq 2de^{-\frac{\log^2 d}{8}}$ for any $\boldsymbol{x} \in \mathbb{R}^d$ with $\|\boldsymbol{x}\|_2 = 1$, termed as $(\log d, 2de^{-\frac{\log^2 d}{8}})$-balanced, hence no dimension carries too much of the ℓ_2 norm of the vector $\boldsymbol{x}$. The second block $\boldsymbol{H}\boldsymbol{D}_2$ ensures that vectors are close to orthogonal. The third block $\boldsymbol{H}\boldsymbol{D}_3$ controls the capacity of the entire structured transform by providing a vector of parameters. SORF is not an unbiased estimator of the Gaussian kernel, but it satisfies an asymptotic unbiased property

$$\left|\mathbb{E}\left[\text{SORF}\right] - e^{-\tau^2/2}\right| \leq \frac{6\tau}{\sqrt{d}}\,.$$

ROM [80]: It generalizes SORF to the form

$$\boldsymbol{W}_{\text{ROM}} = \frac{\sqrt{d}}{\varsigma}\prod_{i=1}^{t}\boldsymbol{H}\boldsymbol{D}_i\,,$$

where $\boldsymbol{H}$ can be the normalized Hadamard matrix or the Walsh matrix, and $\boldsymbol{D}_i$ is the Rademacher matrix as defined in SORF. Theoretical results in [80] show that the ROM estimator achieves variance reduction compared to RFF. Interestingly, odd values of t yield better results than even t. This provides an explanation for why SORF chooses $t = 3$.

LP-RFF [45]: It attempts to quantize RFF with the Gaussian kernel under a memory budget, i.e., mapping each s-dimensional random feature $z_p(\boldsymbol{x}) = \sqrt{2/s}\cos(\boldsymbol{W}_{\text{RFF}}\boldsymbol{x}) \in [-\sqrt{2/s}, \sqrt{2/s}]$ to an s-dimensional low precision vector with b bits via a stochastic rounding scheme. They divide the interval $[-\sqrt{2/s}, \sqrt{2/s}]$ into 2^b-1 equal-sized sub-intervals and randomly round each value $\sqrt{2/s}\cos(\boldsymbol{\omega}_i\boldsymbol{x})$ to either the top or bottom of the corresponding

sub-interval. Strictly speaking, this method does not belong to data-independent algorithms. But we put it here for ease of description as this approach directly quantizes RFF. More importantly, a new insight demonstrated by this method is that, under the same memory budget, random features based algorithms achieve better generalization performance than Nyström approximation [93]. Apart from the stochastic quantization scheme used in [45], the authors of [98] employ Lloyd-Max quantization with a smaller number of bits.

From the above description, one can find that orthogonalization is a typical operation for variance reduction, e.g., ORF/SORF/ROM. Here we take the Gaussian kernel as an example to illustrate insights of such scheme. By sampling $\{\boldsymbol{\omega}_i\}_{i=1}^s \sim \mathcal{N}(\mathbf{0}, \varsigma^{-2}\boldsymbol{I}_d)$, the used Gaussian distribution is isotropic and only depends on the norm $\|\boldsymbol{\omega}\|_2$ instead of $\boldsymbol{\omega}$. The used orthogonal operator makes the direction of $\boldsymbol{\omega}_i$ orthogonal to each other (that means more uniform) while retaining its norm unchanged[5], which leads to decrease the randomness in Monte Carlo sampling, and thus achieve variance reduction effect. If we attempt to directly decrease the randomness in Monte Carlo sampling, QMC is a powerful way to achieve this goal and can then be used to kernel approximation. This is another line of random features with variance reduction illustrated as below.

3.2 Quasi-Monte Carlo Sampling

Here we briefly review methods based on quasi-Monte Carlo sampling (QMC) [37], spherical structured feature (SSF) [43], and moment matching (MM) [42]. These three methods achieve a lower variance or approximation error than RFF. Strictly speaking, the later two algorithms do not belong to the quasi-Monte Carlo sampling framework. However, SSF and MM share the same integration formulation with QMC and thus we introduce them here for simplicity.

Classical Monte Carlo sampling generates a sequence of samples randomly and independently, which may lead to an undesired clustering effect and empty spaces between the samples [92]. Instead of fully random samples, QMC [37] outputs low-discrepancy sequences. A typical QMC sequence has a hierarchical structure: the initial points are sampled on a coarse scale whereas the subsequent points are sampled more finely. For approximating a high-dimensional integral, QMC achieves an asymptotic error convergence rate of $\epsilon = \mathcal{O}((\log s)^d/s)$, which is faster than the $\mathcal{O}(s^{-1/2})$ rate of Monte Carlo. Note however that QMC often requires s to be exponential in d for the improvement to manifest.

QMC [37]: It assumes that $p(\cdot)$ factorizes with respect to the dimensions, i.e., $p(\boldsymbol{x}) = \prod_{j=1}^d p_j(x_j)$, where each $p_j(\cdot)$ is a univariate density function. QMC generally transforms an integral on $\mathbb{R}^d$ in Eq. (4) to one on the unit cube $[0,1]^d$ as

$$k(\boldsymbol{x}-\boldsymbol{x}') = \int_{[0,1]^d} \exp\big(\mathrm{i}(\boldsymbol{x}-\boldsymbol{x}')^\top \Phi^{-1}(\boldsymbol{t})\big)\mathrm{d}\boldsymbol{t}\,, \tag{13}$$

where $\Phi^{-1}(\boldsymbol{t}) = \big(\Phi_1^{-1}(t_1), \cdots, \Phi_d^{-1}(t_d)\big) \in \mathbb{R}^d$ with Φ_j being the cumulative distribution function (CDF) of p_j. Accordingly, by generating a low *discrepancy* sequence $\boldsymbol{t}_1, \boldsymbol{t}_2, \cdots, \boldsymbol{t}_s \in [0,1]^d$, the random frequencies can be constructed by $\boldsymbol{\omega}_i = \Phi^{-1}(\boldsymbol{t}_i)$. The corresponding transformation matrix for QMC is

$$\boldsymbol{W}_{\text{QMC}} = [\Phi^{-1}(\boldsymbol{t}_1), \Phi^{-1}(\boldsymbol{t}_2), \cdots, \Phi^{-1}(\boldsymbol{t}_s)]^\top \in \mathbb{R}^{s\times d}. \tag{14}$$

5. In fact, while orthogonalization only makes the direction of $\{\boldsymbol{\omega}_i\}_{i=1}^s$ more uniform, one can make the length $\|\boldsymbol{\omega}_i\|_2$ uniform by sampling from the cumulative distribution function of $\|\boldsymbol{\omega}\|_2$.

SSF [43]: It improves the space and time complexities of QMC for approximating shift- and rotation-invariant kernels. SSF generates points $\{\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_s\}$ asymptotically uniformly distributed on the sphere $\mathbb{S}^{d-1}$, and construct the transformation matrix as

$$\boldsymbol{W}_{\text{SSF}} = [\Phi^{-1}(t)\boldsymbol{v}_1, \Phi^{-1}(t)\boldsymbol{v}_2, \cdots, \Phi^{-1}(t)\boldsymbol{v}_s]^\top \in \mathbb{R}^{s\times d}\,,$$

where $\Phi^{-1}(t)$ uses the one-dimensional QMC point. The structure matrix $\boldsymbol{V} := [\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_s] \in \mathbb{S}^{(d-1)\times s}$ has the form

$$\boldsymbol{V} = \frac{1}{\sqrt{d/2}} \begin{bmatrix} \operatorname{Re}\boldsymbol{F}_\Lambda & -\operatorname{Im}\boldsymbol{F}_\Lambda \\ \operatorname{Im}\boldsymbol{F}_\Lambda & \operatorname{Re}\boldsymbol{F}_\Lambda \end{bmatrix} \in \mathbb{R}^{d\times s}\,,$$

where $\boldsymbol{F}_\Lambda \in \mathbb{C}^{\frac{d}{2}\times\frac{s}{2}}$ consists of a subset of the rows of the discrete Fourier matrix $\boldsymbol{F} \in \mathbb{C}^{\frac{s}{2}\times\frac{s}{2}}$. The selection of $\frac{d}{2}$ rows from $\boldsymbol{F}$ is done by minimizing the discrete Riesz 0-energy [99] such that the points spread as evenly as possible on the sphere.

MM [42]: It also uses the transformation matrix in Eq. (14), but generates a d-dimensional uniform sampling sequence $\{\boldsymbol{t}_i\}_{i=1}^s$ by a moment matching scheme instead of using a low discrepancy sequence as in QMC. In particular, the transformation matrix is

$$\boldsymbol{W}_{\text{MM}} = [\widetilde{\Phi}^{-1}(\boldsymbol{t}_1), \widetilde{\Phi}^{-1}(\boldsymbol{t}_2), \cdots, \widetilde{\Phi}^{-1}(\boldsymbol{t}_s)]^\top \in \mathbb{R}^{s\times d}, \tag{15}$$

where one uses moment matching to construct the vectors $\widetilde{\Phi}^{-1}(\boldsymbol{t}_i) = \tilde{\boldsymbol{A}}^{-1}(\Phi^{-1}(\boldsymbol{t}_i) - \tilde{\boldsymbol{\mu}})$ with the sample mean $\tilde{\boldsymbol{\mu}} = \frac{1}{s}\sum_{i=1}^s \Phi^{-1}(\boldsymbol{t}_i)$ and the square root of the sample covariance matrix $\tilde{\boldsymbol{A}}$ satisfying $\tilde{\boldsymbol{A}}\tilde{\boldsymbol{A}}^\top = \operatorname{Cov}(\Phi^{-1}(\boldsymbol{t}_i) - \tilde{\boldsymbol{\mu}})$.

To achieve the target of variance reduction, both orthogonalization in Monte Carlo sampling and QMC based algorithms share the similar principle, namely, generating random features that are as independent/uniform as possible. To be specific, QMC and MM are able to generate more uniform data points to avoid undesirable *clustering* effect, see Figure 1 in [37]. Likewise, SSF aims to generate asymptotically uniformly distributed points on the sphere $\mathbb{S}^{d-1}$, which attempts to encode more information with fewer random features, and thus allows for variance reduction. In sampling theory, QMC can be further improved by an sub-grouped based rank-one lattice construction [100] for computational efficiency, which can be used for the subsequent kernel approximation.

3.3 Quadrature based Methods

Quadrature based methods build on a long line of work on numerical quadrature for estimating integrals. In quadrature methods, the weights are often non-uniform, and the points are usually selected using *deterministic* rules including Gaussian quadrature (GQ) [26], [101] and sparse grids quadrature (SGQ) [26]. Deterministic rules can be extended to their stochastic versions. For example, Munkhoeva et al. [27] explore the stochastic spherical-radial (SSR) rule [102], [103] in kernel approximation. Below we briefly review these methods.

GQ [26]: It assumes that the kernel function k factorizes with respect to the dimensions and the corresponding distribution $p(\boldsymbol{\omega}) = p([\omega^{(1)}, \omega^{(2)}, \ldots, \omega^{(d)}]^\top)$ in Eq. (4) is sub-Gaussian. Therefore, the d-dimenionsal integral in Eq. (4) can be factorized as

$$k(\boldsymbol{x}-\boldsymbol{x}') = \prod_{j=1}^d \bigg(\int_{-\infty}^{\infty} p_j(\omega^{(j)}) \exp\big(\mathrm{i}\omega^{(j)}(x^{(j)} - x'^{(j)})\big)\mathrm{d}\omega^{(j)}\bigg). \tag{16}$$

Since each of the factors is a one-dimensional integral, we can approximate them using a one-dimensional quadrature rule. For example, one may use Gaussian quadrature [101] with orthogonal polynomials:

$$\int_{-\infty}^{\infty} p(\omega)\exp(\mathrm{i}\omega(x-x'))\mathrm{d}\omega \approx \sum_{j=1}^{L} a_j \exp\left(\mathrm{i}\gamma_j^\top (x-x')\right), \tag{17}$$

where L is the accuracy level and each γ_j is a univariate point associated with the weight a_j. For a third-point rule with the points $\{-\hat{p}_1, 0, \hat{p}_1\}$ and their associated weights $(\hat{a}_1, \hat{a}_0, \hat{a}_1)$, the transformation matrix $\boldsymbol{W}_{\mathrm{GQ}} \in \mathbb{R}^{s\times d}$ has entries W_{ij} following the distribution

$$\Pr\left(W_{ij}=\pm\hat{p}_1\right)=\hat{a}_1,\ \Pr\left(W_{ij}=0\right)=\hat{a}_0,\ \forall i\in[s], j\in[d].$$

In general, the univariate Gaussian quadrature with L quadrature points is exact for polynomials up to $(2L-1)$ degrees. The multivariate Gaussian quadrature is exact for all polynomials of the form $\omega_1^{i_1}\omega_2^{i_2}\cdots\omega_d^{i_d}$ with $1\le i_j\le 2L-1$; however the total number of points $s=L^d$ scales exponentially with the dimension d and thus this method suffers from the curse of dimensionality.

SGQ [26]: To alleviate the curse of dimensionality, SGQ uses the Smolyak rule [104] to decrease the needed number of points. Here we consider the third-degree SGQ using the symmetric univariate quadrature points $\{-\hat{p}_1, 0, \hat{p}_1\}$ with weights $(\hat{a}_1, \hat{a}_0, \hat{a}_1)$:

$$k(\boldsymbol{x},\boldsymbol{x}')\approx(1-d+d\hat{a}_0)\,g(\boldsymbol{0})+\hat{a}_1\sum_{j=1}^{d}\left[g\left(\hat{p}_1\boldsymbol{e}_j\right)+g\left(-\hat{p}_1\boldsymbol{e}_j\right)\right],$$

where the function $g(\boldsymbol{\omega}) := \sigma(\boldsymbol{\omega}^\top\boldsymbol{x})\sigma(\boldsymbol{\omega}^\top\boldsymbol{x}')$ is given by Eq. (6), and $\boldsymbol{e}_i$ is the d-dimensional standard basis vector with the i-th element being 1. The corresponding transformation matrix is

$$\boldsymbol{W}_{\mathrm{SGQ}}=[\boldsymbol{0}_d,\hat{p}_1\boldsymbol{e}_1,\cdots,\hat{p}_1\boldsymbol{e}_d,-\hat{p}_1\boldsymbol{e}_1,\cdots,-\hat{p}_1\boldsymbol{e}_d]^\top\in\mathbb{R}^{(2d+1)\times d},$$

which leads to the explicit feature mapping

$$\varphi(\boldsymbol{x}) = [\hat{a}_0 g(\boldsymbol{0}), \hat{a}_1 g(\boldsymbol{w}_2^\top\boldsymbol{x}),\cdots, g(\boldsymbol{w}_{2d+1}^\top\boldsymbol{x})],$$

where $\boldsymbol{w}_i$ is the i-th row of $\boldsymbol{W}_{\mathrm{SGQ}}$. Note that SGQ generates $2d+1$ points. To obtain a *dimension-adaptive* feature mapping, Dao et al. [26] propose to subsample the points according to the distribution determined by their weights such that the mapping feature dimension is equal to s.

SSR [27]: It transforms Eq. (6) (actually a d-dimensional integral) to a double integral over a hyper-sphere and the real line. Let $\boldsymbol{\omega} = r\boldsymbol{u}$ with $\boldsymbol{u}^\top\boldsymbol{u}=1$ for $r\in[0,\infty)$, we have

$$k(\boldsymbol{x}-\boldsymbol{x}') = \frac{C_d}{2}\int_{\mathbb{S}^{d-1}}\int_{-\infty}^{\infty} e^{-\frac{r^2}{2}}|r|^{d-1} g(r\boldsymbol{u})\mathrm{d}r\mathrm{d}\boldsymbol{u}, \tag{18}$$

where the integrand is $g(\boldsymbol{\omega}) := \sigma(\boldsymbol{\omega}^\top\boldsymbol{x})\sigma(\boldsymbol{\omega}^\top\boldsymbol{x}')$ given in Eq. (6) and $C_d := (2\pi)^{-d/2}$. The inner integral in Eq. (18) can be approximated by stochastic *radial* rules of degree $2l+1$, i.e., $R(g) = \sum_{i=0}^{l}\hat{w}_i\frac{g(\rho_i)+g(-\rho_i)}{2}$. The outer integral over the d-sphere in Eq. (18) can be approximated by stochastic *spherical* rules: $S_{\mathrm{Q}}(g) = \sum_{j=1}^{q}\widetilde{w}_j g\left(\boldsymbol{Q}\boldsymbol{u}_j\right)$, where $\boldsymbol{Q}$ is a random orthogonal matrix and $\widetilde{w}_j$ are stochastic weights whose distributions are such that the rule is exact for polynomials of degree q and gives unbiased estimate for other functions. Combining the above two rules, we have the SSR rule. Accordingly, the transformation matrix of SSR is

$$\boldsymbol{W}_{\mathrm{SSR}} = \boldsymbol{\vartheta}\otimes\begin{bmatrix}(\boldsymbol{QV})^\top\\ -(\boldsymbol{QV})^\top\end{bmatrix}\in\mathbb{R}^{2(d+1)\times d},$$

with $\boldsymbol{\vartheta} = [\vartheta_1,\vartheta_2,\cdots,\vartheta_s]$ and $\boldsymbol{V} = [\boldsymbol{v}_1,\boldsymbol{v}_2,\cdots,\boldsymbol{v}_{d+1}]$, where $\vartheta\sim\chi(d+2)$ and $\{\boldsymbol{v}_i\}_{i=1}^{d+1}$ are the vertices of a unit regular d-simplex, which is randomly rotated by $\boldsymbol{Q}$. To get s features, one may stack $s/(2d+3)$ independent copies of $\boldsymbol{W}$ as suggested by [27]. Finally, the feature mapping by SSR is given by

$$\varphi(\boldsymbol{x}) = [a_0 g(\boldsymbol{0}), a_1 g(\boldsymbol{w}_1^\top\boldsymbol{x}),\cdots,a_s g(\boldsymbol{w}_s^\top\boldsymbol{x})],$$

where $a_0 = \sqrt{1-\sum_{j=1}^{d+1}\frac{d}{\rho_j^2}}$, $a_j = \frac{1}{\rho_j}\sqrt{\frac{d}{2(d+1)}}$ for $j\in[s]$, and w_j is the j-th element of the stacked $\boldsymbol{W}$.

In general, according to Eq. (6), kernel approximation by random features is actually a d-dimensional integration approximation problem in mathematics. Sampling methods and quadrature based rules are two typical classes of approaches for high-dimensional integration approximation. Efforts on quadrature based methods focus on developing a high-accuracy, mesh-free, efficiency rule, e.g., [105], [106]. Note that, if the integrand $g(\boldsymbol{\omega}) := \sigma(\boldsymbol{\omega}^\top\boldsymbol{x})\sigma(\boldsymbol{\omega}^\top\boldsymbol{x}')$ in the integration representation (6) belongs to a RKHS, the above quadrature rules can be termed as kernel-based quadrature, e.g., Bayesian quadrature [107], [108] and leverage-score quadrature [52]. This approach is in essence different from the previously studied quadrature rules in functional spaces, model formulation, and scope of application.

4 Data-dependent algorithms

Data-dependent approaches aim to design/learn the random features using the training data so as to achieve better approximation quality or generalization performance. Based on how the random features are generated, we can group these algorithms into three classes: *leverage score sampling*, *random features selection*, and *kernel learning by random features*.

4.1 Leverage score based sampling

Leverage score based approaches [31], [47], [109] are built on the *importance sampling* framework. Here one samples $\{\boldsymbol{w}_i\}_{i=1}^{s}$ from a distribution $q(\boldsymbol{w})$ that needs to be designed, and then uses the following feature mapping in Eq. (5):

$$\varphi_q(\boldsymbol{x}) = \frac{1}{\sqrt{s}}\left(\sqrt{\frac{p\left(\boldsymbol{w}_1\right)}{q\left(\boldsymbol{w}_1\right)}}e^{-\mathrm{i}\boldsymbol{w}_1^\top\boldsymbol{x}},\cdots,\sqrt{\frac{p\left(\boldsymbol{w}_s\right)}{q\left(\boldsymbol{w}_s\right)}}e^{-\mathrm{i}\boldsymbol{w}_s^\top\boldsymbol{x}}\right)^\top. \tag{19}$$

Consequently, we have the approximation $k(\boldsymbol{x},\boldsymbol{x}') = \mathbb{E}_{\boldsymbol{w}\sim q}[\varphi_q(\boldsymbol{x})^\top\varphi_q(\boldsymbol{x}')] \approx \sum_{i=1}^{s} z_q(\boldsymbol{w}_i,\boldsymbol{x})z_q(\boldsymbol{w}_i,\boldsymbol{x}')$, where $z_q(\boldsymbol{w}_i,\boldsymbol{x}_j) := \sqrt{p(\boldsymbol{w}_i)/q(\boldsymbol{w}_i)}z_p(\boldsymbol{w}_i,\boldsymbol{x}_j)$. Thus, the kernel matrix $\boldsymbol{K}$ can be approximated by $\boldsymbol{K}_q = \boldsymbol{Z}_q\boldsymbol{Z}_q^\top$, where $\boldsymbol{Z}_q := [\varphi_q(\boldsymbol{x}_1),\cdots,\varphi_q(\boldsymbol{x}_n)]^\top \in \mathbb{R}^{n\times s}$. Denoting by $\boldsymbol{z}_{q,\boldsymbol{w}_i}(\boldsymbol{X})$ the i-th column of $\boldsymbol{Z}_q$, we have $\boldsymbol{K} = \mathbb{E}_{\boldsymbol{w}\sim p}[\boldsymbol{z}_{p,\boldsymbol{w}}(\boldsymbol{X})\boldsymbol{z}_{p,\boldsymbol{w}}^\top(\boldsymbol{X})] = \mathbb{E}_{\boldsymbol{w}\sim q}[\boldsymbol{z}_{q,\boldsymbol{w}}(\boldsymbol{X})\boldsymbol{z}_{q,\boldsymbol{w}}^\top(\boldsymbol{X})]$.

To design the distribution q, one makes use of the ridge leverage function [51], [52] in KRR:

$$l_\lambda(\boldsymbol{\omega}_i) = p(\boldsymbol{\omega}_i)\boldsymbol{z}_{p,\boldsymbol{\omega}_i}^\top(\boldsymbol{X})(\boldsymbol{K}+n\lambda\boldsymbol{I})^{-1}\boldsymbol{z}_{p,\boldsymbol{\omega}_i}(\boldsymbol{X}), \tag{20}$$

where λ is the KRR regularization parameter. Define

$$d_{\boldsymbol{K}}^{\lambda} := \int_{\mathbb{R}^d} l_\lambda(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} = \mathrm{tr}\left[\boldsymbol{K}(\boldsymbol{K}+n\lambda\boldsymbol{I})^{-1}\right]. \tag{21}$$

The quantity $d_{\boldsymbol{K}}^{\lambda} \ll n$ determines the number of independent parameters in a learning problem and hence is referred to as the *number of effective degrees of freedom* [110], [111]. With the above notation, the distribution q designed in [51] is given by

$$q(\boldsymbol{\omega}) := \frac{l_\lambda(\boldsymbol{\omega})}{\int l_\lambda(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}} = \frac{l_\lambda(\boldsymbol{\omega})}{d_{\boldsymbol{K}}^{\lambda}} . \tag{22}$$

Compared to standard Monte Carlo sampling for RFF, leverage score sampling requires fewer Fourier features and enjoys nice theoretical guarantees [31], [51] (see the next section for details). Note that $q(\boldsymbol{\omega})$ can be also defined by the integral operator [52], [112] rather than the Gram matrix used above, but we do not strictly distinguish these two cases. The typical leverage score based sampling algorithm for RFF is illustrated in [31] as below.

LS-RFF (Leverage Score-RFF) [31]: It uses a subset of data to approximate the matrix $\boldsymbol{K}$ in Eq. (21) so as to compute $d_{\boldsymbol{K}}^{\lambda}$. LS-RFF needs $\mathcal{O}(ns^2 + s^3)$ time to generate refined random features, which can be used in KRR [31] and SVM [11] for prediction.

SLS-RFF (Surrogate Leverage Score-RFF) [47]: To avoid inverting an $s \times s$ matrix in LS-RFF, SLS-RFF designs a simple but effective surrogate leverage function

$$L_\lambda(\boldsymbol{w}) = p(\boldsymbol{w})\boldsymbol{z}_{p,\boldsymbol{w}}^\top(\boldsymbol{X}) \left(\frac{1}{n^2\lambda}\Big(\boldsymbol{y}\boldsymbol{y}^\top + n\boldsymbol{I}\Big)\right) \boldsymbol{z}_{p,\boldsymbol{w}}(\boldsymbol{X}) , \tag{23}$$

where the additional term $n\boldsymbol{I}$ and the coefficient $1/(n^2\lambda)$ in Eq. (23) ensure that L_λ is a *surrogate* function that upper bounds the function l_λ in Eq. (20). One then samples random features from the *surrogate* distribution $Q(\boldsymbol{w}) = \frac{L_\lambda(\boldsymbol{\omega})}{\int L_\lambda(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}}$, which has the same time complexity $\mathcal{O}(ns^2)$ as RFF. SLS-RFF and can be applied to KRR [47] and Canonical Correlation Analysis [109].

Note that leverage scores sampling is a powerful tool used in sub-sampling algorithms for approximating large kernel matrices with theoretical guarantees, in particular in Nyström approximation. Research on this topic mainly focuses on obtaining fast leverage score approximation due to inversion of an n-by-n kernel matrix, e.g., two-pass sampling [113] (LS-RFF belongs to this), online setting [114], path-following algorithm [81], or developing various surrogate leverage score sampling based algorithms [47], [48], [109].

4.2 Re-weighted random features

Here we briefly review three re-weighted methods: KA-RFF [83] by kernel alignment, KP-RFF [44] by kernel polarization, and CLR-RFF [46] by compressed low-rank approximation.

KA-RFF (Kernel Alignment-RFF) [83]: It pre-computes a large number of random features that are generated by RFF, and then select a subset of them by solving a simple optimization problem based on kernel alignment [115]. In particular, the optimization problem is

$$\max_{\boldsymbol{a}\in\mathcal{P}_J} \sum_{i,j=1}^{n} y_i y_j \sum_{t=1}^{J} a_t z_p\left(\boldsymbol{x}_i, \boldsymbol{\omega}_t\right) z_p\left(\boldsymbol{x}_j, \boldsymbol{\omega}_t\right) , \tag{24}$$

where $J > s$ is the number of the candidate random features by RFF, and $\boldsymbol{a}$ is the weight vector. Here the maximization is over the set of distributions $\mathcal{P}_J := \{\boldsymbol{a} : D_f(\boldsymbol{a}\|\boldsymbol{1}/J) \le c\}$, where $c > 0$ is a pre-specified constant and $D_f(P\|Q) := \int f(\frac{\mathrm{d}P}{\mathrm{d}Q})\mathrm{d}Q$ with $f(t) = t^2 - 1$ is the χ^2-divergence between the distributions P and Q (a special case of the f-divergence). Solving the problem (24) learns a (sparse) weight vector $\boldsymbol{a}$ of the candidate random features, so that the kernel matrix matches the target kernel $\boldsymbol{y}\boldsymbol{y}^\top$. Problem (24) can be efficiently solved via bisection over a scalar dual variable, and an ϵ-suboptimal solution can be found in $\mathcal{O}(J\log(1/\epsilon))$ time.

KP-RFF (Kernel Polarization-RFF) [44]: It first generates a large number of random features by RFF and then selects a subset from them using an energy-based scheme

$$\tilde{S}(\boldsymbol{\omega}) = \frac{1}{n}\sum_{i=1}^{n} y_i z_p(\boldsymbol{x}_i, \boldsymbol{\omega}) .$$

Further, the quantity $(1/J)\sum_{i=1}^{J}\tilde{S}^2(\boldsymbol{\omega}_j)$ can be associated with kernel polarization for $\{\boldsymbol{w}_i\}_{i=1}^{J}$ sampled from $p(\boldsymbol{\omega})$. Accordingly, the top s random features with the top $|\tilde{S}(\cdot)|$ values are selected as the refined random features. This algorithm can in fact be regarded as a version of the kernel alignment method for generating random features.

CLR-RFF (Compression Low Rank-RFF) [46]: It first generates a large number of random features and then selects a subset from them by approximately solving the optimization problem

$$\begin{aligned} \min_{\boldsymbol{a}\in\mathbb{R}^J:\|\boldsymbol{a}\|_0\le s} \frac{1}{n^2}\left\|\boldsymbol{Z}_J\boldsymbol{Z}_J^\top - \widetilde{\boldsymbol{Z}}_J(\boldsymbol{a})\widetilde{\boldsymbol{Z}}_J(\boldsymbol{a})^\top\right\|_{\mathrm{F}}^2 = \\ \mathbb{E}_{i,j\overset{\text{i.i.d.}}{\sim}[J]}\left[\varphi_p(\boldsymbol{x}_i)^\top\varphi_p(\boldsymbol{x}_j) - \widetilde{\varphi}_p(\boldsymbol{x}_i)^\top\widetilde{\varphi}_p(\boldsymbol{x}_j)\right] , \end{aligned} \tag{25}$$

where $\varphi_p(\boldsymbol{x}) \in \mathbb{R}^J$ uses J random features, and $\widetilde{\varphi}_p(\boldsymbol{x})$ is

$$\widetilde{\varphi}_p(\boldsymbol{x}) := \frac{1}{\sqrt{J}}\left[a_1\exp(-\mathrm{i}\boldsymbol{\omega}_1^\top\boldsymbol{x}), \cdots, a_J\exp(-\mathrm{i}\boldsymbol{\omega}_J^\top\boldsymbol{x})\right]^\top ,$$

which leads to $\widetilde{\boldsymbol{Z}}_J(\boldsymbol{a}) = [\widetilde{\varphi}_p(\boldsymbol{x}_1), \widetilde{\varphi}_p(\boldsymbol{x}_2), \cdots, \widetilde{\varphi}_p(\boldsymbol{x}_n)] \in \mathbb{R}^{n\times J}$. We can construct a Monte-Carlo estimate of the optimization objective function in Eq. (25) by sampling some pairs $i, j \overset{\text{i.i.d.}}{\sim} [J]$. Therefore, this scheme focuses on a subset of pairs, instead of the all data pairs, by seeking a sparse weight vector $\boldsymbol{a}$ with only s nonzero elements. The problem of building a small, weighted subset of the data that approximates the full dataset, is known as the *Hilbert coreset construction problem*. It can be approximately solved by greedy iterative geodesic ascent [116] or Frank-Wolfe based methods [117]. Another way to obtain the compact random features is using Johnson-Lindenstrauss random projection [118] instead of the above data-dependent optimization scheme.

4.3 Kernel learning by random features

This class of approaches construct random features using sophisticated learning techniques, e.g., by learning the spectral distribution of kernel from the data.

Representative approaches in this class often involve a *one-stage* or *two-stage* process. The two-stage scheme is common when using random features. It first learns the random features, and then incorporates them into kernel methods for prediction. Actually, the above-mentioned *leverage sampling* and *random features selection* based algorithms employ this scheme. The algorithm proposed in [84] is a typical method for kernel learning by random features. This method first learns a spectral distribution of a kernel via an implicit generative model, and then trains a linear model by these learned features.

One-stage algorithms aim to simultaneously learn the spectral distribution of a kernel and the prediction model by solving a single joint optimization problem or using a spectral inference scheme. For example, Yu et al. [85] propose to jointly optimize

- approximation error
  - $\|k-\tilde{k}\|_\infty$: [9], [30], [49], [50]
  - $\|k-\tilde{k}\|_{L^r}$: [30], [49]
  - Δ-spectral approximation: [51], [97]
  - (Δ_1, Δ_2)-spectral approximation: [45]
- empirical risk: [45], [51]
- expected risk
  - squared loss
    - $\boldsymbol{\omega} \sim p(\boldsymbol{\omega})$: [31], [53]
    - $\boldsymbol{\omega} \sim q(\boldsymbol{\omega})$: [31]
  - Lipschitz continuous
    - $\boldsymbol{\omega} \sim p(\boldsymbol{\omega})$: [31], [119]
    - $\boldsymbol{\omega} \sim q(\boldsymbol{\omega})$: [11], [31], [52]

Figure 3. Taxonomy of theoretical results on random features.

the nonlinear feature mapping matrix $\boldsymbol{W}$ and the linear model with the hinge loss. The associated optimization problem can be solved in an alternating fashion with SGD. In [86], the kernel alignment approach in the Fourier domain and SVM are combined into a unified framework, which can be also solved using an alternating scheme by Langevin dynamics and projection gradient descent. Wilson and Adams [87] construct stationary kernels as the Fourier transform of a Gaussian mixture based on Gaussian process frequency functions. This approach can be extended to learning with Fastfood [88], non-stationary spectral kernel generalization [70], [71], and the harmonizable mixture kernel [89]. Moreover, Oliva et al. [90] propose a nonparametric Bayesian model, in which $p(\boldsymbol{\omega})$ is modeled as a mixture of Gaussians with a Dirichlet process prior. The parameters of the Gaussian mixture and the classifier/regressor model are inferred using MCMC.

5 Theoretical Analysis

In this section, we review a range of theoretical results that center around the two questions mentioned in the introduction and restated below:

1) **Approximation:** how many random features are needed to ensure a high quality estimator in kernel approximation?
2) **Generalization:** how many random features are needed to incur no loss of empirical risk and expected risk in a learning estimator?

Figure 3 provides a taxonomy of representative work on these two questions.

For the approximation error, existing work focuses on $\|k-\tilde{k}\|_\infty$ [9], [30], [49], $\|k-\tilde{k}\|_{L^r}$ with $1 \leq r < \infty$ [49], Δ-spectral approximation [51], [97], and (Δ_1, Δ_2)-spectral approximation [45]. For the empirical risk under the fixed design setting, existing work provides guarantees on the expected in-sample predication error of the KRR estimator based on Δ-spectral approximation bounds [51] and (Δ_1, Δ_2)-spectral approximation bounds [45]. For the expected risk, a series of works investigate the generalization properties of methods based on $p(\boldsymbol{\omega})$-*sampling* or $q(\boldsymbol{\omega})$-*sampling*. These results cover loss functions with/without Lipschitz continuity and apply to e.g. KRR [31], [53] and SVM [11], [32], [52] under different assumptions.

More specifically, Rahimi and Recht [32] provide the earliest result on learning with RFF with Lipschitz continuous loss functions. Their results imply that $\Omega(n)$ random features are sufficient to incur no loss of learning accuracy. This result is improved in [31], which shows that $\Omega(\sqrt{n}\log n)$ random features or even less suffice for the Gaussian kernel. When using the data-dependent sampling $\{\boldsymbol{\omega}_i\}_{i=1}^s \sim q(\boldsymbol{\omega})$, the above results are further improved in [11], [31], [52] under various settings. Note that some results above do not directly apply to the squared loss in KRR, whose Lipschitz parameter is unbounded. For squared losses, Rudi et al. [53] show that $\Omega(\sqrt{n}\log n)$ random features by RFF suffice to achieve a minimax optimal learning rate $\mathcal{O}(1/\sqrt{n})$. A more refined analysis is given in [31] under the $p(\boldsymbol{\omega})$-sampling and $q(\boldsymbol{\omega})$-sampling settings.

Below we discuss the above theoretical work in more details.

5.1 Approximation error

Table 3 summarizes representative theoretical results on the convergence rates, the upper bound of the growing diameter, and the resulting sample complexity under different metrics. Here sample complexity means the number of random features sufficient for achieving a maximum approximation error at most ϵ.

The first result of this kind is given by Rahimi and Recht [9], who use a covering number argument to derive a uniform convergence guarantee as follows. For a compact subset $\mathcal{S}$ of $\mathbb{R}^d$, let $|\mathcal{S}| := \sup_{\boldsymbol{x},\boldsymbol{x}' \in \mathcal{S}} \|\boldsymbol{x}-\boldsymbol{x}'\|_2$ be its diameter and consider the L^∞ error $\|k-\tilde{k}\|_\infty := \sup_{\boldsymbol{x},\boldsymbol{x}' \in \mathcal{S}} |k(\boldsymbol{x},\boldsymbol{x}') - \tilde{k}(\boldsymbol{x},\boldsymbol{x}')|$.

Theorem 4. *[Uniform convergence of RFF [9], [30]] Let $\mathcal{S}$ be a compact subset of $\mathbb{R}^d$ with diameter $|\mathcal{S}|$. Then, for a stationary kernel k and its approximated kernel $\tilde{k}$ obtained by RFF, we have*

$$\Pr\left[\|k-\tilde{k}\|_\infty \geq \epsilon\right] \leq C_d \left(\frac{\varsigma_p |\mathcal{S}|}{\epsilon}\right)^{\frac{2d}{d+2}} \exp\left(-\frac{s\epsilon^2}{4(d+2)}\right),$$

where $\varsigma_p^2 = \mathbb{E}_p[\boldsymbol{\omega}^\top \boldsymbol{\omega}] = \operatorname{tr} \nabla^2 k(0) \in \mathcal{O}(d)$, and $C_d := 2^{\frac{6d+2}{d+2}} \left(\left(\frac{2}{d}\right)^{\frac{d}{d+2}} + \left(\frac{d}{2}\right)^{\frac{2}{d+2}} \right)$ satisfies $C_d \leq 256$ in [9] and is further improved to $C_d \leq 66$ in [30] by optimization balls of radius in covering number.

According to the above theorem by covering number, with $s := \Omega(\epsilon^{-2} d \log(1/\epsilon\delta))$ random features, one can ensure an ϵ uniform approximation error with probability greater than $1-\delta$. This result also applies to dot-product kernels by random Maclaurin feature maps (see [34, Theorem 8]). The quadrature based algorithm [27] follows this proof framework, and achieves the same error bound with a smaller constant than RFF in Theorem 4 by an extra boundedness assumption. Instead, Fastfood [36] on Gaussian kernels achieves $\mathcal{O}(\sqrt{\log(d/\delta)})$ times approximation error than RFF due to estimates for $\boldsymbol{\Gamma H B}_2$ in Eq. (9), which is based on concentration inequalities for Lipschitz continuous functions under the Gaussian distribution.

Different from the above results using Hoeffding's inequality for the covering number bound in their proof, Sriperumbudur and Szabó [49] revisit the above bound by refined technique of McDiarmid's inequality, symmetrization and bound the expectation of Rademacher average by Dudley entropy bound.

Theorem 5 (Theorem 1 in [49])**.** *Under the same assumption of Theorem 4, we have*

$$\Pr\left[\|k-\tilde{k}\|_\infty \geq \frac{h(d,|\mathcal{S}|,\sigma_p) + \sqrt{2\epsilon}}{\sqrt{s}}\right] \leq e^{-\epsilon},$$

where $h(d,|\mathcal{S}|,\sigma_p)$ is an appropriately defined function of d, $|\mathcal{S}|$, and σ_p. For better comparison, the above inequality can

be rewritten as [50]

$$\Pr\left[\|k-\tilde{k}\|_\infty \geq \epsilon\right] \leq [(\sigma_p+1)(2|\mathcal{S}|+1)]^{1024d} \exp\left(-\frac{s\epsilon^2}{2}+\frac{256d}{\log(2|\mathcal{S}|+1)}\right).$$

Theorem 5 shows that $\tilde{k}$ is a consistent estimator of k in the topology of compact convergence as $s \to \infty$ with the convergence rate $\mathcal{O}_p(\sqrt{s^{-1}\log|\mathcal{S}|})$. Consequently, $\mathcal{O}(\epsilon^{-2}\log|\mathcal{S}|)$ random features suffice to achieve an ϵ approximation accuracy. This sample complexity bound scales logarithmically with $|\mathcal{S}|$, which improves upon the $\mathcal{O}(\epsilon^{-2}|\mathcal{S}|^2\log(|\mathcal{S}|/\epsilon))$ bound that follows from [9], [30] (cf. Theorem 4). Apart from the L^∞ error bound, the authors of [49] further derive bounds on the L^r error $\|k-\tilde{k}\|_{L^r} := \left(\int_{\mathcal{S}}\int_{\mathcal{S}}|k(\boldsymbol{x},\boldsymbol{x}')-\tilde{k}(\boldsymbol{x},\boldsymbol{x}')|^r \mathrm{d}\boldsymbol{x}\mathrm{d}\boldsymbol{x}'\right)^{1/r}$ for $1 \leq r < \infty$; see Table 3 for a summary. We remark that the $L^2_{\rho_{\mathcal{X}}}$ error bound is also given in [30], though the rate in [49] is sometimes better in terms of the diameter.

For the Gaussian kernel, the approximation guarantee can be further improved. In particular, the following theorem gives a probability bound independent of d.

Theorem 6 (Theorem 1 in [50])**.** *Under the same assumption of Theorem 4, for the Gaussian kernel k and its approximation $\tilde{k}$ by RFF, we have*

$$\Pr\left[\|k-\tilde{k}\|_\infty \geq \epsilon\right] \leq \frac{3}{s^{1/3}}\left(\frac{|\mathcal{S}|}{\epsilon}\right)^{2/3}\exp\left(-\frac{s\epsilon^2}{12}\right).$$

Avron et al. [51] argue that the above point-wise distances $\|k-\tilde{k}\|_\infty$ or $\|k-\tilde{k}\|_{L^r}$ are not sufficient to accurately measure the approximation quality. Instead, they focus on the following spectral approximation criterion.

Definition 1. *[Δ-spectral approximation [51]] For $0 \leq \Delta < 1$, a symmetric matrix $\boldsymbol{A}$ is a Δ-spectral approximation of another symmetric matrix $\boldsymbol{B}$, if $(1-\Delta)\boldsymbol{B} \preceq \boldsymbol{A} \preceq (1+\Delta)\boldsymbol{B}$, where $A \preceq B$ indicates that $B-A$ is a semi-positive definite matrix.*

According to this definition, $\boldsymbol{Z}\boldsymbol{Z}^\top + \lambda\boldsymbol{I}_n$ is Δ-spectral approximation of $\boldsymbol{K}+\lambda\boldsymbol{I}_n$ if

$$(1-\Delta)(\boldsymbol{K}+\lambda\boldsymbol{I}_n) \preceq \boldsymbol{Z}\boldsymbol{Z}^\top + \lambda\boldsymbol{I}_n \preceq (1+\Delta)(\boldsymbol{K}+\lambda\boldsymbol{I}_n).$$

The follow theorem gives the number of random features s that are sufficient to guarantee Δ-spectral approximation.

Theorem 7 (Theorem 7 in [51])**.** *Let k be a shift-invariant kernel and its associated probability distribution $p(\boldsymbol{\omega})$ (i.e., the Fourier transform of k), $\Delta \leq 1/2$, $\delta \in (0,1)$, and $n_\lambda := n/\lambda$. Assume that $\|\boldsymbol{K}\|_2 \geq \lambda$ and $\{\boldsymbol{\omega}_i\}_{i=1}^s \sim p(\boldsymbol{\omega})$. If the total number of random features satisfies*

$$s \geq \frac{8}{3}\Delta^{-2} n_\lambda \log\left(16 d_{\boldsymbol{K}}^\lambda/\delta\right),$$

then

$$\Pr\left[(1-\Delta)(\boldsymbol{K}+\lambda\boldsymbol{I}_n) \preceq \boldsymbol{Z}\boldsymbol{Z}^\top+\lambda\boldsymbol{I}_n \preceq (1+\Delta)(\boldsymbol{K}+\lambda\boldsymbol{I}_n)\right] \geq 1-16 d_{\boldsymbol{K}}^\lambda \exp\left(\frac{-3s\Delta^2}{8n_\lambda}\right) \geq 1-\delta.$$

Theorem 7 states that $\Omega(n_\lambda \log d_{\boldsymbol{K}}^\lambda)$ random features are sufficient to guarantee Δ-spectral approximation by the matrix Bernstein concentration inequality and effective degree of freedom, where $n_\lambda := n/\lambda$. Under this framework, Choromanski et al. [97, Theorem 5.4] present a non-asymptotic comparison result between RFF and ORF for spectral approximation by virtue of the smallest singular value of $\boldsymbol{K}+n\lambda\boldsymbol{I}$.

Theorem 8 (Theorem 5.4 in [97])**.** *For the Gaussian kernel, let $\widetilde{\Delta}$ be the smallest positive number such that $\widetilde{\boldsymbol{K}}+\lambda n\boldsymbol{I}_n$ is a $\widetilde{\Delta}$-spectral approximation of $\boldsymbol{K}+\lambda n\boldsymbol{I}_n$, where $\widetilde{\boldsymbol{K}}$ is an approximate kernel matrix obtained by RFF or ORF. Then, for any $\epsilon > 0$ we have*

$$\Pr[\widetilde{\Delta} > \epsilon] \leq \frac{B}{\epsilon^2\sigma_{\min}^2},$$

where $B := \mathbb{E}[\|\widetilde{\boldsymbol{K}}-\boldsymbol{K}\|_{\mathrm{F}}^2]$ and $\sigma_{\min}^2$ is the smallest singular value of $\boldsymbol{K}+\lambda n\boldsymbol{I}_n$. In particular, letting B^{ORF} denotes the value of B for the estimator ORF and B^{RFF} for RFF, we have

$$B^{\mathrm{RFF}}-B^{\mathrm{ORF}} = \frac{s-1}{s}\left(\frac{1}{2d}\sum_{i,j=1}^n \frac{\|\boldsymbol{x}_i-\boldsymbol{x}_j\|_2^4}{\varsigma^2}e^{-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}_j\|_2^2}{\varsigma^2}}+\mathcal{O}\Big(\frac{1}{d}\Big)\right).$$

Theorem 8 shows that $B^{\mathrm{RFF}} > B^{\mathrm{ORF}}$ always holds for the Gaussian kernel. To better understand the above upper bound on $\Pr[\widetilde{\Delta} > \epsilon]$, we note that both $\mathbb{V}[\mathrm{RFF}]$ and $\mathbb{V}[\mathrm{ORF}]$ are $\mathcal{O}(1/s)$, hence $B = \mathcal{O}(n^2/s)$. Moreover, since the Gaussian kernel has exponentially decaying eigenvalues (see Assumption 4), we have $\sigma_{\min}^2 = \Omega(n^2\lambda^2)$. Therefore, the upper bound of $\Pr[\widetilde{\Delta} > \epsilon]$ is on the order of $\mathcal{O}(\frac{1}{s\lambda^2})$. With the standard scaling of the regularization parameter $\lambda = n^{-\alpha}$, $\alpha \in (0,1]$, we need $s := \Omega(n^{2\alpha})$ to get a non-trivial upper bound on the probability. When $\alpha = 1/2$, these results for RFF and ORF require $\Omega(n)$ random Fourier features, which is somewhat unsatisfactory [31].

The results in Theorem 7 can be improved if we consider data-dependent sampling, i.e., $\{\boldsymbol{\omega}_i\}_{i=1}^s$ are sampled from the empirical ridge leverage score distribution $q(\boldsymbol{\omega}) = l_\lambda(\boldsymbol{\omega})/d_{\boldsymbol{K}}^\lambda$ in Eq. (22) instead of the standard $p(\boldsymbol{\omega})$.

Theorem 9 (Lemma 6 in [51])**.** *Let k be a shift-invariant kernel associated with the empirical ridge leverage score distribution $q(\boldsymbol{\omega})$ in Eq.* (22)*, $\Delta \leq 1/2$ and $\delta \in (0,1)$. Assume that $\|\boldsymbol{K}\|_2 \geq \lambda$ and $\{\boldsymbol{\omega}_i\}_{i=1}^s \sim q(\boldsymbol{\omega})$. If the total number of random features satisfies*

$$s \geq \frac{8}{3}\Delta^{-2} d_{\boldsymbol{K}}^\lambda \log\left(16 d_{\boldsymbol{K}}^\lambda/\delta\right),$$

then

$$\Pr\left[(1-\Delta)(\boldsymbol{K}+\lambda\boldsymbol{I}_n) \preceq \boldsymbol{Z}\boldsymbol{Z}^\top+\lambda\boldsymbol{I}_n \preceq (1+\Delta)(\boldsymbol{K}+\lambda\boldsymbol{I}_n)\right] \geq 1-16 d_{\boldsymbol{K}}^\lambda \exp\left(\frac{-3s\Delta^2}{8 d_{\boldsymbol{K}}^\lambda}\right) \geq 1-\delta.$$

Theorem 9 shows that if we sample using the ridge leverage function, then $\Omega(d_{\boldsymbol{K}}^\lambda \log d_{\boldsymbol{K}}^\lambda)$ random features, which is less than $\Omega(n_\lambda \log d_{\boldsymbol{K}}^\lambda)$, suffice for spectral approximation of $\boldsymbol{K}$.

The authors of [45] generalize the notion of Δ-spectral approximation to (Δ_1,Δ_2)-spectral approximation.

Definition 2 ((Δ_1,Δ_2)-spectral approximation [45])**.** *For $\Delta_1,\Delta_2 \geq 0$, a symmetric matrix $\boldsymbol{A}$ is a (Δ_1,Δ_2)-spectral approximation of another symmetric matrix $\boldsymbol{B}$, if $(1-\Delta_1)\boldsymbol{B} \preceq \boldsymbol{A} \preceq (1+\Delta_2)\boldsymbol{B}$.*

This definition is motivation by the argument that the quantities Δ_1 and Δ_2 in the upper and lower bounds may have different impact on the generalization performance. Using this definition, Zhang et al. [45] derive the following approximation guarantees

Table 3
Comparison of convergence rates and required random features for kernel approximation error.

Metric	Results	Convergence rate	Upper bound of $\lvert\mathcal{S}\rvert$	Required random features s
$\lVert k-\tilde{k}\rVert_\infty$	Theorem 4 ([9], [30])	$\mathcal{O}_p\left(\lvert\mathcal{S}\rvert\sqrt{\frac{\log s}{s}}\right)$	$\lvert\mathcal{S}\rvert \leq \Omega\left(\sqrt{\frac{s}{\log s}}\right)$	$s \geq \Omega\left(d\epsilon^{-2}\log\frac{\lvert\mathcal{S}\rvert}{\epsilon}\right)$
	Theorem 1 in [49]	$\mathcal{O}_p\left(\sqrt{\frac{\log\lvert\mathcal{S}\rvert}{s}}\right)$	$\lvert\mathcal{S}\rvert \leq \Omega(s^c)$[1]	$s \geq \Omega\left(d\epsilon^{-2}\log\lvert\mathcal{S}\rvert\right)$
	Theorem 1 in [50] (Gaussian kernels)	$\mathcal{O}_p\left(\sqrt{\frac{\log\lvert\mathcal{S}\rvert}{s}}\right)$	$\lvert\mathcal{S}\rvert \leq \Omega(s^c)$	$s \geq \Omega\left(\epsilon^{-2}\log\lvert\mathcal{S}\rvert\right)$
$\lVert k-\tilde{k}\rVert_{L^r}\ (1\leq r<\infty)$	Corollary 2 in [49]	$\mathcal{O}_p\left(\lvert\mathcal{S}\rvert^{\frac{2d}{r}}\sqrt{\frac{\log\lvert\mathcal{S}\rvert}{s}}\right)$	$\lvert\mathcal{S}\rvert \leq \Omega\left((\frac{s}{\log s})^{\frac{r}{4d}}\right)$	$s \geq \Omega\left(d\epsilon^{-2}\log\lvert\mathcal{S}\rvert\right)$
$\lVert k-\tilde{k}\rVert_{L^r}\ (2\leq r<\infty)$	Theorem 3 in [49]	$\mathcal{O}_p\left(\lvert\mathcal{S}\rvert^{\frac{2d}{r}}\sqrt{\frac{1}{s}}\right)$	$\lvert\mathcal{S}\rvert \leq \Omega\left(s^{\frac{r}{4d}}\right)$	$s \geq \Omega\left(d\epsilon^{-2}\log\lvert\mathcal{S}\rvert\right)$
Δ-spectral approximation	Theorem 7 in [51]	$\mathcal{O}_p\left(\sqrt{\frac{n_\lambda}{s}}\right)$	-	$s \geq \Omega(n_\lambda \log d_{\boldsymbol{K}}^\lambda)$
	Theorem 5.4 in [97] (Gaussian kernels)	$\mathcal{O}_{\text{RFF/ORF}}\left(\frac{1}{s\lambda^2}\right)$	-	$s \geq \Omega(n^{2\alpha})$
	Lemma 6 in [51]	$\mathcal{O}_q\left(\sqrt{\frac{d_{\boldsymbol{K}}^\lambda}{s}}\right)$	-	$s \geq \Omega(d_{\boldsymbol{K}}^\lambda \log d_{\boldsymbol{K}}^\lambda)$
(Δ_1,Δ_2)-spectral approximation	Theorem 2 in [45]	$\mathcal{O}_{\text{LP}}\left(\sqrt{\frac{n_\lambda}{s}}\right)$ [2]	-	$s \geq \Omega(n_\lambda \log d_{\boldsymbol{K}}^\lambda)$

[1] c is some constant satisfying $0<c<1$.

[2] LP denotes that $\{\boldsymbol{\omega}_i\}_{i=1}^s$ are obtained by RFF and then are quantized to a Low-Precision b-bit representation; see [45].

when one quantizes each random Fourier feature $\boldsymbol{\omega}_i$ to a low-precision b-bit representation, which allows more features to be stored in the same amount of space.

Theorem 10 (Theorem 2 in [45])**.** *Let $\widetilde{\boldsymbol{K}}$ be an s-features b-bit LP-RFF approximation of a kernel matrix $\boldsymbol{K}$ and $\delta \in (0,1)$. Assume that $\lVert\boldsymbol{K}\rVert_2 \geq \lambda \geq \delta_b^2 = 2/(2^b-1)^2$ and define $a := 8\,\mathrm{Tr}(\boldsymbol{K}+\lambda\boldsymbol{I}_n)^{-1}(\boldsymbol{K}+\delta_b^2\boldsymbol{I}_n)$. For $\Delta_1 \leq 3/2$ and $\Delta_2 \in [\frac{\delta_b^2}{\lambda}, \frac{3}{2}]$, if the total number of random features satisfies*

$$s \geq \frac{8}{3} n_\lambda \max\left\{\frac{2}{\Delta_1}, \frac{2}{\Delta_2 - \delta_b^2/\lambda}\right\} \log\left(\frac{a}{\delta}\right),$$

then

$$\begin{aligned}&\Pr\left[(1-\Delta_1)(\boldsymbol{K}+\lambda\boldsymbol{I}_n) \preceq \tilde{\boldsymbol{K}}+\lambda\boldsymbol{I}_n \preceq (1+\Delta_2)(\boldsymbol{K}+\lambda\boldsymbol{I}_n)\right]\\ &\geq 1 - a\left[\exp\left(\frac{-3s\Delta_1^2}{4n_\lambda(1+2/3\Delta_1)}\right)\right.\\ &\quad\left.+\exp\left(\frac{-3s(\Delta_2-\delta_b^2/\lambda)^2}{4n_\lambda(1+2/3(\Delta_2-\delta_b^2/\lambda))}\right)\right].\end{aligned}$$

Theorem 10 shows that when the quantization noise is small relative to the regularization parameter, using low precision has minimal impact on the number of features required for the (Δ_1,Δ_2)-spectral approximation. In particular, as $s\to\infty$, Δ_1 converges to zero for any precision b, whereas Δ_2 converges to a value upper bounded by δ_b^2/λ. If $\delta_b^2/\lambda \ll \Delta_2$, using b-bit precision has negligible effect on the number of features required to attain this Δ_2 see Table 3 for a summary.

5.2 Risk and generalization property

The above results on approximation error are a means to an end. More directly related to the learning performance is understanding generalization properties of random features based algorithms. To this end, a series of work study the generalization properties of algorithms based on $p(\boldsymbol{\omega})$-sampling and $q(\boldsymbol{\omega})$-sampling. Under different assumptions, theoretical results have been obtained for loss functions with/without Lipschitz continuity and for learning tasks including KRR [31], [53] and SVM [11], [32], [52]. Apart from supervised learning with random features, results on randomized nonlinear component analysis refer to [10], random features with matrix sketching [120], doubly stochastic gradients scheme [94], statistical consistency [121], [122].

5.2.1 Assumptions

Before we detail these theoretical results, we summarize the standard assumptions imposed in existing work. Some assumptions are technical, and thus familiarity with statistical learning theory (see Section 2.1) would be helpful. We organize these assumptions in four categories as shown in Figure 4, including i) the existence of f_ρ (Assumption 1) and its stronger version (Assumption 8); ii) quality of random features (Assumptions 2, 6, 7); iii) noise conditions (Assumptions 3, 9, 10); iv) eigenvalue decay (Assumptions 4, 5).

We first state three basic assumptions, which are needed in all of the (regression) results to be presented.

Assumption 1 (Existence [53], [123])**.** *In regression task, we assume $f_\rho \in \mathcal{H}$.*

Note that since we consider a potentially infinite dimensional RKHS $\mathcal{H}$, possibly universal [124], the existence of the target function f_ρ is not automatic. However, if we restrict to a bounded subspace of $\mathcal{H}$, i.e., $\mathcal{H}_R = \{f\in\mathcal{H} : \lVert f\rVert \leq R\}$ with $R<\infty$ fixed a prior, then a minimizer of the risk $\mathcal{E}(f)$ always exists as long as $\mathcal{H}_R$ is not universal. If f_ρ exists, then it must lie in a ball of some radius $R_{\rho,\mathcal{H}}$. The results in this section do not require prior knowledge of $R_{\rho,\mathcal{H}}$ and they hold for any finite radius.

Assumption 2 (Random features are bounded and continuous [53])**.** *For the shift-invariant kernel k, we assume that $\varphi(\boldsymbol{\omega}^\top\boldsymbol{x})$ in Eq. (6) is continuous in both variables and bounded, i.e., there exists $\kappa \geq 1$ such that $\lvert\varphi(\boldsymbol{\omega}^\top\boldsymbol{x})\rvert < \kappa$ for all $\boldsymbol{x}\in\mathcal{X}$ and $\boldsymbol{\omega}\in\mathbb{R}^d$.*

- i) regression: $f_\rho \in \mathcal{H}$ (**Ass. 1**) $\Leftarrow$ source condition (**Ass. 8**)
- ii) quality of random features
  - bounded and continuous (**Ass. 2**)
  - $q(\boldsymbol{\omega})$-sampling: compatibility condition (**Ass. 6**) $\Leftarrow$ optimized distribution (**Ass. 7**)
- iii) noise condition
  - regression: boundedness on y (**Ass. 3**)
  - classification: Massart's low noise condition (**Ass. 9**) $\Leftarrow$ separation condition (**Ass. 10**)
- iv) eigenvalue decays assumption (**Ass. 4**)
  - exponential decay
  - polynomial decay and harmonic decay $\Leftrightarrow$ capacity condition (**Ass. 5**)

Figure 4. Relationship between the needed assumptions. The notation $A \Leftarrow B$ means that B is a stronger assumption than A.

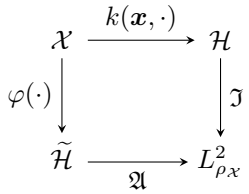


Figure 5. Maps between various spaces.

Assumption 3 (Bernstein's condition [124], [125]). *For any $\boldsymbol{x} \in \mathcal{X}$, we assume $\mathbb{E}\left[|y|^b \mid \boldsymbol{x}\right] \leq \frac{1}{2} b! \varsigma^2 B^{b-2}$ when $b \geq 2$.*

This noise condition is weaker than the boundedness on y. It is satisfied when y is bounded, sub-Gaussian, or sub-exponential. In particular, if $y \in [-\frac{b}{2}, \frac{b}{2}]$ almost surely with $b > 0$, then Assumption 3 is satisfied with $\varsigma = B = b$.

The above three assumptions are needed in all theoretical results for regression presented in this section, so we omit them when stating these results. We next introduce several additional assumptions, which are needed in some of the theoretical results.

Eigenvalue Decay Assumptions: The following assumption, which characterizes the "size" of the RKHS $\mathcal{H}$ of interest, is often discussed in learning theory.

Assumption 4 (Eigenvalue decays [111]). *A kernel matrix $\boldsymbol{K}$ admit the following three types of eigenvalue decays: 1)* Geometric/exponential decay*: $\lambda_i(\boldsymbol{K}) \propto n \exp(-i^{1/c})$, which leads to $d^\lambda_{\boldsymbol{K}} \lesssim \log(R_0/\lambda)$; 2)* Polynomial decay*: $\lambda_i(\boldsymbol{K}) \propto n i^{-2a}$, which implies $d^\lambda_{\boldsymbol{K}} \lesssim (1/\lambda)^{1/2a}$; 3)* Harmonic decay*: $\lambda_i(\boldsymbol{K}) \propto n/i$, which results in $d^\lambda_{\boldsymbol{K}} \lesssim (1/\lambda)$.*

We give some remarks on the above assumption. For shift-invariant kernels, if the RKHS is small, the eigenvalues of the kernel matrix $\boldsymbol{K}$ often admit a fast decay. Consequently, functions in the RKHS are smooth enough that a good prediction performance can be achieved. On the other hand, if the RKHS is large and the eigenvalues decay slowly, then functions in the RKHS are not smooth, which would lead to a sub-optimal error rate for prediction. It can be linked to the integral operator [123], [124] characterizing the hypothesis space, defined as $\Sigma : L^2_{\rho_\mathcal{X}} \to L^2_{\rho_\mathcal{X}}$ such that

$$(\Sigma g)(\boldsymbol{x}) = \int_\mathcal{X} k(\boldsymbol{x}, \boldsymbol{x}') g(\boldsymbol{x}') \mathrm{d}\rho_\mathcal{X}(\boldsymbol{x}'), \quad \forall g \in L^2_{\rho_\mathcal{X}}\,.$$

It is clear that the operator Σ is self-adjoint, positive definite, and trace-class when $k(\cdot,\cdot)$ is continuous. This operator can be represented as $\Sigma = \mathfrak{I}\mathfrak{I}^*$ in terms of the inclusion operator $\mathfrak{I} : \mathcal{H} \to L^2_{\rho_\mathcal{X}},\ (\mathfrak{I} f) = f$. Here $\mathfrak{I}^*$ is the adjoint of $\mathfrak{I}$ and is given by

$$\mathfrak{I}^* : L^2_{\rho_\mathcal{X}} \to \mathcal{H},\ (\mathfrak{I}^* f)(\cdot) = \int_\mathcal{X} k(\boldsymbol{x}, \cdot) f(\boldsymbol{x}) \mathrm{d}\rho_\mathcal{X}\,,$$

due to the self-adjoint property of the Hilbert spaces $L^2_{\rho_\mathcal{X}}$ and $\mathcal{H}$ [122]. With s random features, the inclusion operator $\mathfrak{I}$ can be approximated by the operator $\mathfrak{A} : \widetilde{\mathcal{H}} \to L^2_{\rho_\mathcal{X}},\ (\mathfrak{A}\boldsymbol{\beta}) = \langle \varphi(\cdot), \boldsymbol{\beta} \rangle_{\widetilde{\mathcal{H}}},\ \forall \beta \in \mathbb{R}^s$. Figure 5 presents the relationship between various spaces under different operators.

The integral operator Σ plays a significant role in characterizing the hypothesis space. In particular, the decay rate of the spectrum of Σ quantifies the capacity of the hypothesis space in which we search for the solution. This capacity in turn determines the number of random features required for accurate learning. Rudi and Rosasco [53] consider the following assumption on Σ.

Assumption 5 (Capacity condition [123], [126]). *There exist $Q > 0$ and $\gamma \in [0, 1]$ such that for any $\lambda > 0$, we have*

$$\mathcal{N}(\lambda) := \mathrm{tr}\left((\Sigma + \lambda I)^{-1}\Sigma\right) \leq Q^2 \lambda^{-\gamma}\,. \tag{26}$$

The effective dimension $\mathcal{N}(\lambda)$ [110] measures the "size" of the RKHS, and is in fact the operator form of $d^\lambda_{\boldsymbol{K}}$ in Eq. (21). Assumption 5 holds if the eigenvalues λ_i of Σ decay as $i^{-1/\gamma}$, which corresponds to the eigenvalue decay of $\boldsymbol{K}$ in Assumption 4 with $\gamma := 1/(2a)$ [127]. The case $\gamma = 0$ is the more benign situation, whereas $\gamma = 1$ is the worst case.

Quality of Random Features: Here we introduce several technical assumptions on the quality of random features. The leverage score in Eq. (20) admits the operator form

$$\mathcal{F}_\infty(\lambda) := \sup_{\boldsymbol{\omega}} \left\| (\Sigma + \lambda I)^{-1/2} \varphi(\boldsymbol{x}) \right\|^2_{L^2_{\rho_\mathcal{X}}},\ \forall \lambda > 0\,,$$

which is also called as the *maximum random features dimension* [53]. By defintion we always have $\mathcal{N}(\lambda) \leq \mathcal{F}_\infty(\lambda)$. Roughly speaking, when the random features are "good", it is easy to control their leverage scores in terms of the decay of the spectrum of Σ. Further, fast learning rates using fewer random features can be achieved if the features are *compatible* with the data distribution in the following sense.

Assumption 6 (Compatibility condition [53]). *With the above definition of $\mathcal{F}_\infty(\lambda)$, assume that there exist $\varrho \in [0, 1]$, and $F > 0$ such that $\mathcal{F}_\infty(\lambda) \leq F \lambda^{-\varrho}, \forall \lambda > 0$.*

It always holds that $\mathcal{F}_\infty(\lambda) \leq \kappa^2 \lambda^{-1}$ when z is uniformly bounded by κ. So the worst case is $\varrho = 1$, which means that the random features are sampled in a problem independent way. The favorable case is $\varrho = \gamma$, which means that $\mathcal{N}(\lambda) \leq \mathcal{F}_\infty(\lambda) \leq \mathcal{O}(n^{-\alpha\gamma})$. In [11], the authors consider the following assumption.

Assumption 7 (Optimized distribution [11]). *The feature mapping $z(\boldsymbol{\omega}, \boldsymbol{x})$ is called* optimized *if there is a small constant λ_0 such that for any $\lambda \leq \lambda_0$, $\mathcal{F}_\infty(\lambda) \leq \mathcal{N}(\lambda) = \sum_{i=1}^\infty \frac{\lambda_i(\Sigma)}{\lambda_i(\Sigma)+\lambda}$.*

Under the previous definitions, Assumption 7 holds only when $\mathcal{F}_\infty(\lambda) = \mathcal{N}(\lambda)$. This assumption is stronger than the

compatibility condition in Assumption 6. Note that Assumption 7 is satisfied when sampling from $q(\boldsymbol{\omega})$.

Source condition on f_ρ: The following assumption states that f_ρ has some desirable regularity properties.

Assumption 8 (Source condition [53], [128])**.** *There exist $1/2 \leq r \leq 1$ and $g \in L^2_{\rho_\mathcal{X}}$ such that $f_\rho(\boldsymbol{x}) = (\Sigma^r g)(\boldsymbol{x})$ almost surely.*

Since Σ is a compact positive operator on $L^2_{\rho_\mathcal{X}}$, its r-th power Σ^r is well defined for any $r > 0$.[6] Assumption 8 imposes a form of regularity/sparsity of f_ρ, which requires the expansion of f_ρ on the basis given by the integral operator Σ. Note that this assumption is more stringent than the existence of f_ρ in $\mathcal{H}$. The latter is equivalent to Assumption 8 with $r = \frac{1}{2}$ (the worst case), in which case $f_\rho \in \mathcal{H}$ need not have much regularity/sparsity.

Noise Condition: The following two assumptions on noise are considered in random features for classification.

Assumption 9 (Massart's low noise condition [11])**.** *There exists $V \geq 2$ such that*

$$\left|\mathbb{E}_{(\boldsymbol{x},y)\sim\rho}[y|\boldsymbol{x}]\right| \geq 2/V\,.$$

Assumption 10 (Separation condition [11])**.** *The points in $\mathcal{X}$ can be collected into two sets according to their labels as follows*

$$X_1 := \{\boldsymbol{x} \in \mathcal{X} : \mathbb{E}[y|\boldsymbol{x}] > 0\}\,,$$
$$X_{-1} := \{\boldsymbol{x} \in \mathcal{X} : \mathbb{E}[y|\boldsymbol{x}] < 0\}\,.$$

For $i \in \{\pm 1\}$, the distance of a point $\boldsymbol{x} \in X_i$ to the set X_{-i} is denoted by $\Delta(\boldsymbol{x})$. We say that the data distribution satisfies a separation condition if there exists $\Delta > 0$ such that $\rho_X(\Delta(\boldsymbol{x}) < c) = 0$.

The above two assumptions, both controlling the noise level in the labels, can be cast under into a unified framework [131] as follows. Define the *regression function* $\eta(\boldsymbol{x}) = \mathbb{E}[y|X = \boldsymbol{x}]$ in binary classification problems. The Massart's low noise condition means that there exists $h \in (0, 1]$ such that for $|\eta(\boldsymbol{x})| \geq h$ for all $\boldsymbol{x} \in \mathcal{X}$. Here h characterizes the level of noise in classification problems. If small h is small, then $\eta(\boldsymbol{x})$ is close to zero, in which case correct classification is difficult. Massart's condition can be extended to the following more flexible condition known as Tsybakov's low noise assumption [131]. This assumption stipulates that there exists a constant $C > 0$ such that for all sufficiently small $t > 0$, we have

$$\Pr\left(\{\boldsymbol{x} \in \mathcal{X} : |2\eta(\boldsymbol{x}) - 1| \leq t\}\right) \leq C \cdot t^q\,,$$

for some $q > 0$. The separation condition in Assumption 10 is an extreme case of the Tsybakov's noise assumption with $q = \infty$. It is clear that noise-free distributions satisfy this separation assumption, since the conditional probability η is bounded away from $1/2$.

5.2.2 Squared loss in KRR

In this section, we review theoretical results on the generalization properties of KRR with squared loss and random features, for both the $p(\boldsymbol{\omega})$-sampling (data-independent) and $q(\boldsymbol{\omega})$-sampling (data-dependent) settings. Table 4 summarizes these results for the excess risk in terms of the key assumptions imposed, the learning rates, and the required number of random features.

We begin with the remarkable result by Rudi and Rosasco [53]. They are among the first to show that under some mild assumptions and appropriately chosen parameters, $\Omega(\sqrt{n}\log n)$ random features suffice for KRR to achieve minimax optimal rates.

Theorem 11 (Generalization bound; Theorem 3 in [53])**.** *Suppose that Assumption 8 (source condition) holds with $r \in [\frac{1}{2}, 1]$, Assumption 6 (compatibility) holds with $\varrho \in [0, 1]$, and Assumption 5 (capacity) holds with $\gamma \in [0, 1]$. Assume that $n \geq n_0$ and choose $\lambda := n^{\frac{1}{2r+\gamma}}$. If the number of random features satisfies*

$$s \geq c_0 n^{\frac{\alpha+(2r-1)(1+\gamma-\alpha)}{2r+\gamma}} \log\frac{108\kappa^2}{\lambda\delta}\,,$$

then the excess risk of $\tilde{f}_{\boldsymbol{z},\lambda}$ can be upper bounded as

$$\mathcal{E}\left(\tilde{f}_{\boldsymbol{z},\lambda}\right) - \mathcal{E}\left(f_\rho\right) = \left\|\tilde{f}_{\boldsymbol{z},\lambda} - f_\rho\right\|^2_{L^2_{\rho_\mathcal{X}}} \leq c_1 \log^2\frac{18}{\delta} n^{-\frac{2r}{2r+\gamma}}\,,$$

where c_0, c_1 are constants independent of (n, λ, δ), and n_0 does not depends on n, λ, f_ρ, or ρ.

Theorem 11 unifies several results in [53] that impose different assumptions. The simplest result is Theorem 1 in [53], which only requires the three basic Assumptions 1–3 on existence, boundedness and continuity, corresponding to the the worst case of Theorem 11 with $\varrho = \gamma = 1$ and $r = 1/2$. In this case, by choosing $\lambda = n^{-1/2}$, we require $\Omega(\sqrt{n}\log n)$ random features to achieve the minimax convergence rate $\mathcal{O}(n^{-1/2})$; also see Table 4.

A more refined result is given in Theorem 2 in [53], which accounts for the capacity of the RKHS and the regularity of f_ρ, as quantified by the parameters $\gamma \in [0, 1]$ (Assumption 5) and $r \in [\frac{1}{2}, 1]$ (Assumption 8), respectively. Under these conditions and choosing $\lambda := n^{-\frac{1}{2r+\gamma}}$, we require $\Omega\big(n^{\frac{1+\gamma(2r-1)}{2r+\gamma}}\log n\big)$ random features to achieve the convergence rate $\mathcal{O}\big(n^{-\frac{2r}{2r+\gamma}}\big)$. Note that $\gamma = 1$ is the worst case, where the eigenvalues of $\boldsymbol{K}$ have the slowest decay, and $\gamma = 1/(2a) \in (0, 1)$ means that the eigenvalues follow a polynomial decay $\lambda_i \propto ni^{-2a}$. Table 4 presents this result with $\gamma := 1/(2a)$ for better comparison with the other results.

The above two results apply to the standard RFF setting with data-independent sampling. When $\{\boldsymbol{\omega}_i\}_{i=1}^s$ are sampled from a data-dependent distribution satisfying the compatibility condition in Assumption 6 with $\varrho \in [0, 1]$, then Theorem 3 in [53] provide an improved result. In this case, by choosing $\lambda := n^{-\frac{1}{2r+\gamma}}$, we require $\Omega\big(n^{\frac{\varrho+(1+\gamma-\varrho)(2r-1)}{2r+\gamma}}\log n\big)$ random features to achieve the convergence rate $\mathcal{O}\big(n^{-\frac{2r}{2r+\gamma}}\big)$.

If the compatibility condition is replaced by the stronger Assumption 7 (optimized distribution), satisfied by $q(\boldsymbol{\omega})$-sampling, the work [31] derives an improved bound that is the sharpest to date. Below we state a general result from [31] that covers both $p(\boldsymbol{\omega})$- and $q(\boldsymbol{\omega})$-sampling.

Theorem 12 (Theorem 1 in [31])**.** *Suppose that the regularization parameter λ satisfies $0 \leq n\lambda \leq \lambda_1$. We consider two sampling schemes.*

- *$\{\boldsymbol{\omega}_i\}_{i=1}^s \sim p(\boldsymbol{\omega})$: if $s \geq (5z_0^2/\lambda)\log(16d^\lambda_{\boldsymbol{K}}/\delta)$ and $|z(\boldsymbol{\omega}, \boldsymbol{x})| \leq z_0$,*
- *$\{\boldsymbol{\omega}_i\}_{i=1}^s \sim q(\boldsymbol{\omega})$: if $s \geq 5d^\lambda_{\boldsymbol{K}}\log\left(16d^\lambda_{\boldsymbol{K}}/\delta\right)$,*

then for $0 < \delta < 1$, with probability $1 - \delta$, the excess risk of $\tilde{f}_{\boldsymbol{z},\lambda}$ can be upper bounded as

$$\left\|\widetilde{f}_{\boldsymbol{z},\lambda} - f_\rho\right\|^2_{L^2_{\rho_\mathcal{X}}} \leq 2\lambda + \mathcal{O}(1/\sqrt{n}) + \mathcal{E}\big(f_{\boldsymbol{z},\lambda}\big) - \mathcal{E}\left(f_\rho\right)\,, \quad (27)$$

6. A more general condition ($r > 0$) is often considered in approximation theory; see [129], [130].

Table 4
Comparison of learning rates and required random features for expected risk with the squared loss function.

sampling scheme	Results	key assumptions	eigenvalue decays	λ	learning rates	required s
$\{\boldsymbol{\omega}_i\}_{i=1}^s \sim p(\boldsymbol{\omega})$	[53, Theorem 1]	-	-	$n^{-\frac{1}{2}}$	$\mathcal{O}_p\left(n^{-\frac{1}{2}}\right)$	$s \geq \Omega(\sqrt{n}\log n)$
	[53, Theorem 2]	source condition	i^{-2t}	$n^{-\frac{2t}{1+4rt}}$	$\mathcal{O}_p\left(n^{-\frac{4rt}{1+4rt}}\right)$	$s \geq \Omega(\frac{2t+2r-1}{1+4rt}\log n)$
			$1/i$	$n^{-\frac{1}{2r+1}}$	$\mathcal{O}_p\left(n^{-\frac{2r}{2r+1}}\right)$	$s \geq \Omega(n^{\frac{2r}{2r+1}}\log n)$
	[31, Corollary 2]	-	$e^{-\frac{1}{c}i}$	$n^{-\frac{1}{2}}$	$\mathcal{O}_p\left(n^{-\frac{1}{2}}\right)$	$s \geq \Omega(\sqrt{n}\log n)$
			i^{-2t}	$n^{-\frac{1}{2}}$	$\mathcal{O}_p\left(n^{-\frac{1}{2}}\right)$	$s \geq \Omega(\sqrt{n}\log n)$
			$1/i$	$n^{-\frac{1}{2}}$	$\mathcal{O}_p\left(n^{-\frac{1}{2}}\right)$	$s \geq \Omega(\sqrt{n}\log n)$
$\{\boldsymbol{\omega}_i\}_{i=1}^s \sim q(\boldsymbol{\omega})$	[53, Theorem 3]	source condition; compatibility condition	i^{-2t}	$n^{-\frac{2t}{1+4rt}}$	$\mathcal{O}_q\left(n^{-\frac{4rt}{1+4rt}}\right)$	$s \geq \Omega(\frac{\varrho+(2r-1)(2t+1-2t\varrho)}{1+4rt}\log n)$
			$1/i$	$n^{-\frac{1}{2r+1}}$	$\mathcal{O}_q\left(n^{-\frac{2r}{2r+1}}\right)$	$s \geq \Omega(n^{\frac{2r}{2r+1}}\log n)$
	[31, Corollary 1]	optimized distribution	$e^{-\frac{1}{c}i}$	$n^{-\frac{1}{2}}$	$\mathcal{O}_q\left(n^{-\frac{1}{2}}\right)$	$s \geq \Omega(\log^2 n)$
			i^{-2t}	$n^{-\frac{1}{2}}$	$\mathcal{O}_q\left(n^{-\frac{1}{2}}\right)$	$s \geq \Omega(n^{1/(2t)}\log n)$
			$1/i$	$n^{-\frac{1}{2}}$	$\mathcal{O}_q\left(n^{-\frac{1}{2}}\right)$	$s \geq \Omega(\sqrt{n}\log n)$

where we recall that $\mathcal{E}\left(f_{\boldsymbol{z},\lambda}\right) - \mathcal{E}\left(f_\rho\right)$ is the excess risk of standard KRR with an exact kernel (see Section 2).

Remark: A sharper convergence rate can be achieved if the Rademacher complexity used in [31] is substituted by the local Rademacher complexity [132], see [133] for details.

For $p(\boldsymbol{\omega})$-sampling, Theorem 12 improves on the results of [53] under the exponential and polynomial decays. Specifically, if $\{\boldsymbol{\omega}_i\}_{i=1}^s \sim p(\boldsymbol{\omega})$, Theorem 12 requires $s \propto 1/\lambda \log d_{\boldsymbol{K}}^\lambda$. Specialized to the exponential decay case, this result requires $\Omega(\sqrt{n}\log\log n)$ random features to achieve an $\mathcal{O}(n^{-1/2})$ learning rate, which is an improvement compared to [53] with $\Omega(\sqrt{n}\log n)$ random features.

For $q(\boldsymbol{\omega})$-sampling, Theorem 12 shows that if $\lambda = n^{-1/2}$, then $s \propto d_{\boldsymbol{K}}^\lambda \log d_{\boldsymbol{K}}^\lambda$ random features is sufficient to incurs no loss in the expected risk if KRR, with a minimax learning rate $\mathcal{O}(n^{-1/2})$. Corollaries of this result under three different regimes of eigenvalue decay are summarized in Table 4.

Carratino et al. [134] extend the result of [53] to the setting where KRR is solved by stochastic gradient descent (SGD). They show that under the basic Assumptions 1–3 and some mild conditions for SGD, $\Omega(\sqrt{n})$ random features suffice to achieve the minimax learning rate $\mathcal{O}(n^{-1/2})$. This result matches those for standard KRR with an exact kernel [135]. The above results can be improved if in addition the source condition in Assumption 8 holds, in which case $\Omega(n^{\frac{1+\alpha(2r-1)}{2r+\alpha}})$ random features suffice to achieve an $\mathcal{O}(n^{-\frac{2r}{2r+\alpha}})$ learning rate.

The work in [136] shows that if the randomized feature map is bounded (which is weaker than Assumption 2), then we have the following out-of-sample bound

$$\mathcal{E}(\widetilde{f_{\boldsymbol{z},\lambda}}) - \mathcal{E}(f_{\boldsymbol{z},\lambda}) \leq \mathcal{O}\left(\frac{1}{s\lambda}\right).$$

If we choose $\lambda := n^{-1/2}$, then $\Omega(n)$ random features are sufficient to ensure an $\mathcal{O}(n^{-1/2})$ rate in the out-of-sample bound.

5.2.3 Lipschitz continuous loss function

In this section, we consider loss functions ℓ that are Lipschitz continuous. Examples include the hinge loss in SVM and the cross-entropy loss in kernel logistic regression. Table 5 summarizes several existing results for such loss functions in terms of the learning rate and the required number of random features. We briefly discuss these results below and refer the readers to the cited work for the precise theorem statements.

If $\{\boldsymbol{\omega}_i\}_{i=1}^s \sim p(\boldsymbol{\omega})$, i.e., under the standard RFF setting with data-independent sampling, we have the following results.

- Theorem 1 in [32] shows that the excess risk converges at a certain $\mathcal{O}(n^{-1/2})$ rate with $\Omega(n\log n)$ random features.
- Corollary 4 in [31] shows that with $\lambda \in \mathcal{O}(1/n)$ and $\Omega\big((1/\lambda)\log d_{\boldsymbol{K}}^\lambda\big)$ random features, the excess risk of $\tilde{f}_{\boldsymbol{z},\lambda}$ can be upper bounded by

$$\mathcal{E}(\tilde{f}_{\boldsymbol{z},\lambda}) - \mathcal{E}\left(f_\rho\right) \leq \mathcal{O}\left(1/\sqrt{n}\right) + \mathcal{O}(\sqrt{\lambda}).$$

  The above bound scales with $\sqrt{\lambda}$, which is different from the bound in Eq. (27) for the squared loss. Therefore, for Lipschitiz continuous loss functions, we need to choose a smaller regularization parameter $\lambda \in \mathcal{O}(1/n)$ to achieve the same $\mathcal{O}(n^{-1/2})$ convergence rate. Also note that as before we can bound $d_{\boldsymbol{K}}^\lambda$ under the three types of eigenvalue decay.

If $\{\boldsymbol{\omega}_i\}_{i=1}^s \sim q(\boldsymbol{\omega})$, i.e., under the data-dependent sampling setting, we have the following results.

- For SVM with random features, under the optimized distribution in Assumption 7 and the low noise condition in Assumption 9, Theorem 1 in [11] provides bounds on the learning rates and the required number of random features. This result is improved in [11, Theorem 2] if we consider the stronger separation condition in Assumption 10. Details can be found in Table 5.
- In Section 4.5 in [52] and Corollary 3 in [31], it is shown that if Assumption 7 holds, then the excess risk of $\tilde{f}_{\boldsymbol{z},\lambda}$ converges at an $\mathcal{O}(n^{-1/2})$ rate with $\Omega(d_{\boldsymbol{K}}^\lambda \log d_{\boldsymbol{K}}^\lambda)$ random features, if we choose $\lambda \in \mathcal{O}(1/n)$.

Table 5
Comparison of learning rates and required random features for expected risk with a Lipschitz continuous loss function.

sampling scheme	Results	key assumptions	eigenvalue decay	λ	learning rates	required s
$\{\boldsymbol{\omega}_i\}_{i=1}^s \sim p(\boldsymbol{\omega})$	[32, Theorem 1]	-	-	-	$\mathcal{O}_p\left(n^{-\frac{1}{2}}\right)$	$s \geq \Omega(n \log n)$
	[31, Corollary 4]	-	$e^{-\frac{1}{c}i}$	$\frac{1}{n}$	$\mathcal{O}_p\left(n^{-\frac{1}{2}}\right)$	$s \geq \Omega(n \log \log n)$
			i^{-2t}	$\frac{1}{n}$	$\mathcal{O}_p\left(n^{-\frac{1}{2}}\right)$	$s \geq \Omega(n \log n)$
			$1/i$	$\frac{1}{n}$	$\mathcal{O}_p\left(n^{-\frac{1}{2}}\right)$	$s \geq \Omega(n \log n)$
$\{\boldsymbol{\omega}_i\}_{i=1}^s \sim q(\boldsymbol{\omega})$	[11, Theorem 1]	low noise condition	$e^{-\frac{1}{c}i}$	$\frac{1}{n}$	$\mathcal{O}_q(\frac{1}{n}\log^{c+2} n)$	$s \geq \Omega(\log^c n \log\log^c n)$
		optimized distribution	i^{-2t}	$n^{-\frac{t}{1+t}}$	$\mathcal{O}_q\left(n^{-\frac{t}{1+t}}\log n\right)$	$s \geq \Omega(n^{\frac{1}{1+t}}\log n)$
			$1/i$	$\frac{1}{n}$	$\mathcal{O}_q\left(n^{-\frac{1}{2}}\right)$	$s \geq \Omega(n \log n)$
	[11, Theorem 2]	separation condition optimized distribution	$e^{-\frac{1}{c}i}$	n^{-2c^2}	$\mathcal{O}_q(\frac{1}{n}\log^{2c+1} n \log\log n)$	$s \geq \Omega(\log^{2c} n \log\log n)$
	[52, Section 4.5] [31, Corollary 3]	optimized distribution	$e^{-\frac{1}{c}i}$	$\frac{1}{n}$	$\mathcal{O}_q\left(n^{-\frac{1}{2}}\right)$	$s \geq \Omega(\log^2 n)$
			i^{-2t}	$\frac{1}{n}$	$\mathcal{O}_q\left(n^{-\frac{1}{2}}\right)$	$s \geq \Omega(n^{1/(2t)}\log n)$
			$1/i$	$\frac{1}{n}$	$\mathcal{O}_q\left(n^{-\frac{1}{2}}\right)$	$s \geq \Omega(n \log n)$

There is an abnormal but common experiment phenomenon on kernel approximation and risk generalization, that is, a higher kernel approximation quality does not always translate to better generalization performance, see the discussion in [27], [45], [51]. Understanding this inconsistency between approximation quality and generalization performance is an important open problem in this topic. Here we present a preliminary result for KRR: a better approximation quality cannot guarantee a lower generalization risk, see Proposition 1 as below, with proof deferred to Appendix A.

Proposition 1. *Given the target function f_ρ and the original kernel matrix $\boldsymbol{K}$, consider two random features based algorithms* A1 *and* A2 *yielding two approximated kernel matrices $\widetilde{\boldsymbol{K}}_1$ and $\widetilde{\boldsymbol{K}}_2$, and their respective KRR estimators $\tilde{f}_{\boldsymbol{z},\lambda}^{(\mathrm{A1})}$ and $\tilde{f}_{\boldsymbol{z},\lambda}^{(\mathrm{A2})}$. Then for a new sample $\boldsymbol{x}$, even if $\|\boldsymbol{K}-\widetilde{\boldsymbol{K}}_1\| \leq \|\boldsymbol{K}-\widetilde{\boldsymbol{K}}_2\|$ holds in some norm metric, there exists one case for the excess risk such that*

$$\mathcal{E}[\tilde{f}_{\boldsymbol{z},\lambda}^{(\mathrm{A1})}(\boldsymbol{x})] - \mathcal{E}[f_\rho(\boldsymbol{x})] \geq \mathcal{E}[\tilde{f}_{\boldsymbol{z},\lambda}^{(\mathrm{A2})}(\boldsymbol{x})] - \mathcal{E}[f_\rho(\boldsymbol{x})]\,.$$

Remark: Our proof is geometric by constructing a counter-example. It requires that the kernel admits (at least) polynomial decay, which holds for the common-used Gaussian kernel and could be further relaxed for the existence of the proof.

5.3 Results for nonlinear component analysis

In addition to supervised learning problems such as classification and regression, random features can also be used in unsupervised learning, e.g., randomized nonlinear component analysis. Here we give an overview of the results for this problem.

The authors of [10] propose to use random features to approximate the kernel matrix in kernel Principal Component Analysis (KPCA) and kernel Canonical Correlation Analysis (KCCA). They show that the approximate kernel matrix converges to the true one in operator norm at a rate of $\mathcal{O}(n\sqrt{\log n/s})$. More precisely, $s = \mathcal{O}((\log n)^2/\epsilon^2)$ suffices to ensure that $\|\widetilde{\boldsymbol{K}}-\boldsymbol{K}\|_2 \leq \epsilon n$ with the probability $1-1/n$. Their algorithm takes $\mathcal{O}(ns^2+nsd)$ time to construct feature functions and $\mathcal{O}(s^2+sd)$ space to store the feature functions and covariance matrix. Ghashami et al. [120] combine random features with matrix sketching for KPCA. For finding the top-ℓ principal components, they improve the time and space complexities to $\mathcal{O}(nsd+n\ell s)$ and $\mathcal{O}(sd+\ell s)$, respectively. Xie et al. [94] propose to use the doubly stochastic gradients scheme to accelerate KPCA. The authors of [121] investigate the statistical consistency of KPCA with random features. They show that the top-ℓ eigenspace of the empirical covariance matrix in $\widetilde{\mathcal{H}}$ converges to the covariance operator in $\mathcal{H}$ at the rate of $\mathcal{O}(1/\sqrt{n}+1/\sqrt{s})$. Therefore, $\Omega(n)$ random features are required to guarantee a $\mathcal{O}(1/\sqrt{n})$ rate. Ullah et al. [122] instead pose KPCA as a stochastic optimization problem and show that the empirical risk minimizer (ERM) in the random feature space converges in objective value at an $\mathcal{O}(1/\sqrt{n})$ with $\Omega(\ell\sqrt{n}\log n)$ random features.

Table 6
Dataset statistics.

datasets	d	#traing	#test	random split	scaling
ijcnn1	22	49,990	91,701	no	-
EEG	14	7,490	7,490	yes	mapstd
cod-RNA	8	59,535	157,413	no	mapstd
covtype	54	290,506	290,506	yes	minmax
magic04	10	9,510	9,510	yes	minmax
letter	16	12,000	6,000	no	minmax
skin	3	122,529	122,529	yes	minmax
a8a	123	22,696	9,865	no	-
MNIST	784	60,000	10,000	no	minmax
CIFAR-10	3072	50,000	10,000	no	-
MNIST-8M	784	8,100,000	10,000	no	-

6 Experiments

In this section, we empirically evaluate the kernel approximation and classification performance of representative random features algorithms on several benchmark datasets. All experiments are implemented in MATLAB and carried out on a PC with Intel® i7-8700K CPU (3.70 GHz) and 64 GB RAM. The source code of our implementation can be found in http://www.lfhsgre.org.

6.1 Experimental settings

Kernel: We choose the popular Gaussian kernel, zero/first-order arc-cosine kernels, and polynomial kernels for experiments.

i) Gaussian kernel:

$$k(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|_2^2}{2\varsigma^2}\right), \tag{28}$$

where the kernel width parameter ς is tuned via 5-fold inner cross validation over a grid of $\{0.01, 0.1, 1, 10, 100\}$.

To evaluate the Gaussian kernel, we conduct the following representative algorithms for comparison: RFF [9], ORF [24], SORF [24], ROM [80], Fastfood [36], QMC [37], SSF [43], GQ [26], and LS-RFF [31]. These algorithms include both data-independent and data-dependent approaches and involve a variety of techniques including Monte Carlo and quasi-Monte Carlo sampling, quadrature rules, variance reduction, and computational speedup using structural/circulant matrices.

ii) arc-cosine kernels: Different from Gaussian kernels and polynomial kernels, the designed arc-cosine kernels [60] can be closely connected to neural networks, which include feature spaces that mimic the sparse, nonnegative, distributed representations of single-layer threshold networks. The used zeroth order kernel is given explicitly by

$$k(\boldsymbol{x}, \boldsymbol{x}') = 1 - \frac{\theta}{\pi},$$

which corresponds to the Heaviside step function $\sigma(\boldsymbol{\omega}^\top \boldsymbol{x}) = \frac{1}{2}(1 + \mathrm{sign}(\boldsymbol{\omega}^\top \boldsymbol{x}))$ in Eq. (6). The first order kernel is

$$k(\boldsymbol{x}, \boldsymbol{x}') = \frac{1}{\pi}\|\boldsymbol{x}\|_2\|\boldsymbol{x}'\|_2 \left(\sin\theta + (\pi - \theta)\cos\theta\right),$$

which corresponds to the ReLU activation function $\sigma(\boldsymbol{\omega}^\top \boldsymbol{x}) = \max\{0, \boldsymbol{\omega}^\top \boldsymbol{x}\}$ in Eq. (6).

Here we consider the zero/first-order arc-cosine kernel and compare these ten algorithms (used for Gaussian kernel approximation) as well. Note that, the theoretical foundation behind random features, Bocher's theorem, is invalid to arc-cosine kernels. Thankfully, according to the formulation of arc-cosine kernels admitting in Eq. (6), the Monte Carlo sampling (e.g., RFF) is able to used for arc-cosine kernel approximation. In this case, the remaining algorithms, e.g., ORF, QMC, and Fastfood, on various sampling strategies, can be still applicable to arc-cosine kernels, at least in the algorithmic aspect.

iii) Polynomial kernel: This is a widely used family of dot product kernels given by

$$k(\boldsymbol{x}, \boldsymbol{x}') = (1 + \langle \boldsymbol{x}, \boldsymbol{x}' \rangle)^b,$$

where b is the order. In our experiments, the order is set to $b = 2$. Note that, different from Gaussian kernels and arc-cosine kernels, polynomial kernels admit neither the Bochner's theorem nor the sampling formulation in Eq. (6), so classical random features based algorithms are applicable to arc-cosine kernels but still invalid to polynomial kernels even though both of them are dot-product. As a result, algorithms for polynomial kernel approximation are often totally different. In this survey, we include three representative approaches for evaluation, including Random Maclaurin (RM) [34], Tensor Sketch (TS) [73], and Tensorized Random Projection (TRP) [74].

Datasets: We consider eight non-image benchmark datasets, two representative image datasets, and a ultra-large scale dataset for evaluation. Table 6 gives an overview of these datasets including the number of feature dimension, training samples, test data, training/test split, and the normalization scheme. These eight non-image benchmark datasets can be downloaded from https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/ or the UCI Machine Learning Repository[7]. Some datasets include a training/test partition, denoted as "no" in the random split column. For the other datasets, we randomly pick half of the data for training and the rest for testing, denoted as "yes" in the random split column. There are two typical normalization schemes used in these datasets: "mapstd" and "minmax". The "mapstd" scheme sets each sample's mean to 0 and deviation to 1, while the "minmax" scheme is a standard min-max scaling operation mapping the samples to the bounded set $[0, 1]^d$. Two representative image datasets are the *MNIST* handwritten digits dataset [137] and the *CIFAR10* natural image classification dataset [138], summarized in the last two rows in Table 6. The MNIST dataset contains 60,000 training samples and 10,000 test samples, each of which is a 28×28 gray-scale image of a handwritten digit from 0 to 9. Here the "minmax" normalization scheme means that each pixel value is divided by 255. The CIFAR10 dataset consists of 60,000 color images of size $32 \times 32 \times 3$ in 10 categories, with 50,000 for training and 10,000 for test. Besides, apart from medium/large scale datasets in our experiments, we also evaluate the compared approaches on a ultra-large scale dataset *MNIST 8M* [139], which is derived from the *MNIST* dataset by random deformations and translations. It shares the same number of feature dimension and test data with the *MNIST* dataset, but has 8,100,000 training data.

Evaluation metrics: We evaluate the performance of all the compared algorithms in terms of approximation error, time cost, and test accuracy. We use $\|\boldsymbol{K} - \widetilde{\boldsymbol{K}}\|_{\mathrm{F}}/\|\boldsymbol{K}\|_{\mathrm{F}}$ as the error metric for kernel approximation. A small error indicates a high approximation quality. To compute the approximation error, we randomly sample 1,000 data points to construct the sub-feature matrix and the sub-kernel matrix. We record the time cost of each algorithm on generating feature mappings. The kernel width ς in the Gaussian kernel is tuned by five-fold cross validation over the grid $\{0.01, 0.1, 1, 10, 100\}$. The regularization parameter λ in ridge linear regression and the balance parameter in liblinear are tuned via 5-fold inner cross validation on a grid of $\{10^{-8}, 10^{-6}, 10^{-4}, 10^{-3}, 10^{-2}, 0.05, 0.1, 0.5, 1, 5, 10\}$ and $\{0.01, 0.1, 1, 10, 100\}$, respectively. For the sake of computational efficiency, we conduct a relatively coarse hyper-parameter tuning. Nevertheless, a refined hyper-parameter search might result in better classification performance. The random features dimension s in our experiments takes value in $\{2d, 4d, 8d, 16d, 32d\}$. All experiments are repeated 10 times and we report the average approximation error, average classification accuracy with their respective standard deviations as well as the time cost for generating random features.

6.2 Results for the Gaussian Kernel

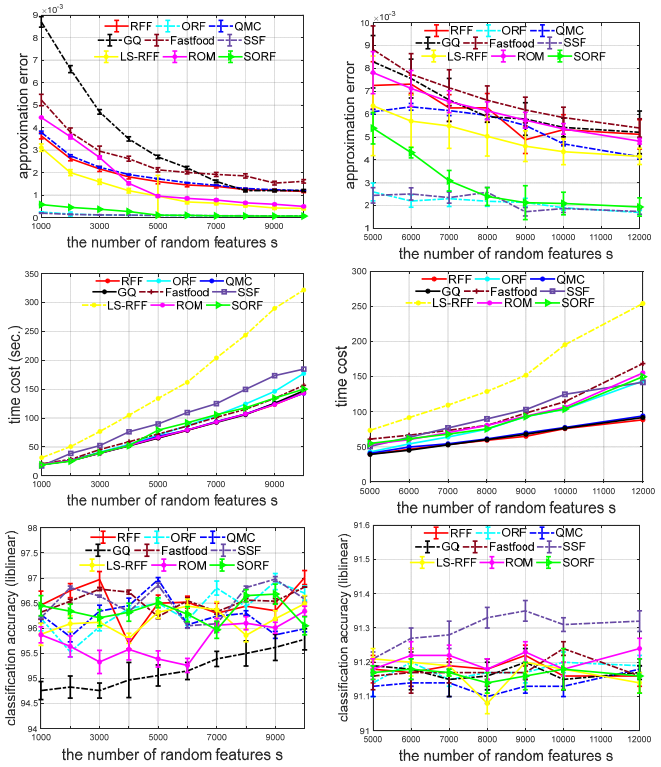
6.2.1 Results on non-image benchmark datasets

Here we test various random features based algorithms, including RFF [9], ORF [24], SORF [24], ROM [80], Fastfood [36], QMC [37], SSF [43], GQ [26], LS-RFF [31] for kernel approximation and then combine these algorithms with lr/liblinear for classification on eight non-image benchmark datasets, refer to Appendix B.1 for details. Here we summarize the best performing algorithm on each dataset in terms of the approximation quality and classification accuracy in Table 7, where we distinguish the small s case (i.e.,

7. https://archive.ics.uci.edu/ml/datasets.html.

Table 7
Results statistics on several classification datasets. The best algorithm on each dataset is given in two cases: low dimensional (i.e., $s = 2d, 4d$) and high dimensional (i.e., $s = 16d, 32d$) according to approximation quality, test accuracy in linear regression or liblinear. The notation "-" means that there is no *statistically significant* difference in the performance of most algorithms.

datasets	approximation		lr		liblinear	
	small s	large s	small s	large s	small s	large s
ijcnn1	SSF	SORF, QMC, ORF	-	-	Fastfood	-
EEG	SSF	ORF	-	-	-	-
cod-RNA	SSF	-	-	-	-	-
covtype	ORF	-	-	-	-	-
magic04	SSF	SSF, ORF, QMC, ROM	-	-	-	-
letter	SSF	SSF, ORF	-	-	-	-
skin	SSF, ROM	QMC	-	-	-	-
a8a	-	-	-	-	SSF	-



(a) MNIST (b) CIFAR10

Figure 6. Approximation error, time cost, and test accuracy of various algorithms with liblinear on two image classification datasets.

$s = 2d$ or $s = 4d$) and the large s case (i.e., $s = 16d$ or $s = 32d$). The notation "-" therein means that there is no *statistically significant* difference in the performance of most algorithms.

In terms of approximation error, we find that SSF, ORF, and QMC achieve promising approximation performance in most cases. Recall that the goal of using random features is to find a finite-dimensional (embedding) Hilbert space to approximate the original infinite-dimensional RKHS so as to preserve the inner product. To achieve this goal, SSF, QMC, and ORF are based on a similar principle, namely, generating random features that are as independent/complete as possible to reduce the randomness in sampling. Regarding to SSF, we find that SSF performs well under the small s case, but the significant improvement does not hold for the large s case. This might be because, a few points can be adequate in SSF, additional points (i.e., a larger s) may have a small marginal benefit in variance reduction under the large s setting. Consequently, the approximation error of SSF sometimes stays almost the same with a larger number of random features. QMC and ORF seek for variance reduction on random features. Nevertheless, they often work well in the large s case. As demonstrated by the expression for variance of ORF [24] and convergence rate in QMC [37], this theoretical result is consistent with the numerical performance of ORF and QMC, which may explain the reason why they work better in a large s setting than a small s case.

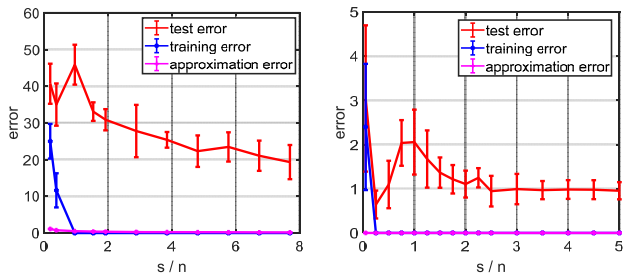
Results on arc-cosine kernels and polynomial kernels can be in Appendix B. Besides, apart from the above used medium/large scale datasets in our experiments, we also evaluate the compared approaches on a ultra-large scale dataset *MNIST 8M* [139] with millions of data. Due to the memory limit, following the doubly stochastic framework [38], we incorporate these random features based approaches under the data streaming setting for the reduction of time and space complexity.

6.2.2 *Classification results on* MNIST *and* CIFAR10

Here we consider the MNIST and CIFAR10 datasets, on which we test these random features based algorithms for kernel approximation and then combine these algorithms with liblinear for image classification. In our experiment, we use the Gaussian kernel[8], whose kernel width ς is tuned by 5-fold cross validation over the grid $\varsigma = [0.01, 0.1, 1, 10, 100]$. For the MNIST database, we directly use the original 784-dimensional feature as the data. For better performance on the CIFAR10 dataset, we use VGG16 with batch normalization [141] pre-trained on ImageNet [142] as a feature extractor. We fine-tune this model on the CIFAR10 dataset with 240 epochs and a mini-batch size 64. The learning rate is initialized at 0.1 and then divided by 10 at the 120-th, 160-th, and 200-th epochs. For each color image, a 4096 dimensional feature vector is obtained from the output of the first fully-connected layer in this fine-tuned neural network.

Figure 6(a) shows the approximation error, the time cost (sec.), and the classification accuracy by liblinear across a range of $s = 1000$ to $s = 10,000$ random features on the MNIST database. We find that ORF and SSF yield the best approximation quality. Despite that most algorithms achieve different approximation errors, there is no significant difference in the test accuracy, which

8. As indicated by [13], [140], (convolutional) NTK generally performs better than Gaussian kernel but it is still non-trivial to obtain a efficient random features mapping for (convolutional) NTK without much loss on prediction.



(a) sonar (low-dimensioanl) (b) MNIST (high-dimensional)

Figure 7. Training error, test error, and approximation error of random features regression with $\lambda = 10^{-8}$ on the *sonar* dataset with $n = 208, d = 60$ and the sub-set of MNIST (class 1 versus class 2) with $n = 200, d = 784$.

corresponds to the results on non-image datasets. Similar results are observed on the CIFAR10 dataset with $s = 5000$ to $s = 12,000$ random features; see Figure 6(b). Note that most algorithms take the similar time cost on generating random features except for the data-dependent algorithm LS-RFF. Several structured based approaches (e.g., Fastfood, SORF, ROM) do not achieve significant reduction on time cost due to the relatively inefficient Matlab built-in function to implement the Walsh-Hadamard transform.

7 Trends: High-dimensional Random Features in Over-parameterized Settings

In the previous sections, we review random features based algorithms and their theoretical results, that works under a fixed d setting with $s \ll n$. Random features based approaches are simple in formulation but enjoy nice empirical validations and theoretical guarantees in kernel approximation and generalization properties. Recently, analysis of over-parameterized models [22], [143], [144], [145], [146] has attracted a lot of attention in learning theory, partly due to the observation of several intriguing phenomena, including capability of fitting random labels, strong generalization performance of overfitted classifiers [19] and double descent in the test error curve [20], [147]. Moreover, Belkin et al. [20], [148] point out that the above phenomena are not unique to deep networks but also exist in random features and random forests. In Figure 7, we report the empirical training error, the test error, and the kernel approximation error of random features regression as a function of s/n on the *sonar* dataset and the *MNIST* dataset [137]. Even with n, d, s only in the hundreds, we can still observe that as s increases, the training error reduces to zero and the approximation error monotonously decreases. However, the test error exhibits double descent, i.e., a phase transition at the *interpolation threshold*: moving away from this threshold on both sides trends to reduce the generalization error. This is somewhat striking as it goes against the conventional wisdom on *bias-variance trade-off* [149]: predictors that generalize well should trade off the model complexity against training data fitting.

The above observations have motivated researchers to build on the elegant theory of random features to provide an analysis of neural networks in the over-parameterized regime. To be specific, RFF can be regarded as a two-layer (large-width) neural network, where the weights in the first layer are chosen randomly/fixed and only the output layer is optimized. This is a typical over-parameterized model if we take $s \gg n$. As such, two-layer neural networks in this regime are more amenable to theoretical analysis as compared to general arbitrary deep networks. This is a potentially fruitful research direction, and one hand, the optimization and generalization of such model have been studied in [22], [150] in deep learning theory. On the other hand, in order to explain the double descent curve of random features in over-parameterized regimes, we often work in a high dimensional setting, which is more subtle than classical results in standard settings, as indicated by recent random matrix theory (RMT) [151], [152], [153]. An intuitive example [154] is, $\|\boldsymbol{K} - \boldsymbol{Z}\boldsymbol{Z}^\top\|_{\mathrm{F}} \to 0$ always hold in low/high dimensions as $s \to \infty$ but $\|\boldsymbol{K} - \boldsymbol{Z}\boldsymbol{Z}^\top\|_2 \to 0$ does not hold for $n, d, s \to \infty$. Accordingly, in this section, we provide an overview on analysis of (high dimensional) random features in over-parameterized setting, especially on double descent. We remark upfront that the random features model on double descent is not the only way for analyzing DNNs. Many other approaches, with different points of views, have been proposed for deep learning theory, but they are out of scope of this survey.

7.1 Results on High Dimensional Random Features in Over-parameterized Setting

Here we briefly introduce the problem setting of high dimensional random features in over-parameterized regimes, and then discuss the techniques used in various studies.

In the basic setting, high dimensional random features often work with least squares regression setting in an asymptotic viewpoint, i.e., $n, d, s \to \infty$ with $d/n \to \psi_1 \in (0, \infty)$ and $s/n \to \psi_2 \in (0, \infty)$, in which overparameterization corresponds to $\psi_2 \geq 1$. The considered data generation model in the basic setting is quite simple. To be specific, the training data is collected in a matrix $\boldsymbol{X} \in \mathbb{R}^{n \times d}$, the rows of which are assumed to be drawn i.i.d from $\mathcal{N}(0, 1)$ or $\mathbb{S}^{d-1}(\sqrt{d})$. The labels are given by a linear ground truth corrupted by some independent additive Gaussian noise: $y_i = f_\rho(\boldsymbol{x}_i) + \varepsilon_i$, where $f_\rho(\boldsymbol{x}) = \langle \boldsymbol{x}, \boldsymbol{\zeta} \rangle$ for a fixed but unknown $\boldsymbol{\zeta}$ and $\varepsilon_i \sim \mathcal{N}(0, 1)$. The transformation matrix under this setting is often taken as the random Gaussian matrix with the ReLU activitation function (recall Eq. (6)). Current approaches employ various data generation schemes and assumptions to obtain a refined analysis beyond double descent under the basic setting. According to these criteria, we summarize the problem setting of various representative approaches in Table 8. In the next, we briefly review the conceptual and technical contributions of underlying approaches on high dimensional random features.

Belkin et al. [162] begin with an one-dimensional (noise-free) version of the random features model, and provide an asymptotic analysis to explain the double descent phenomenon. The subsequent work focuses on the standard random features model under different settings and assumptions. It is clear that, the presence of the nonlinear activation function $\sigma(\cdot)$ makes the random features model intractable to study the related (limiting) spectral distribution. Accordingly, the key issue in this topic mainly focuses on studying random matrices with nonlinear dependencies, e.g., how to disentangle the nonlinear function $\sigma(\cdot)$ by Gaussian equivalence conjecture. Hastie et al. [143] consider the basic setting endowed by a bounded activation function with a standardization condition, i.e., $\mathbb{E}[\sigma(t)] = 0$ and $\mathbb{E}[\sigma(t)^2] = 1$ for $t \sim N(0, 1)$. By establishing asymptotic results on resolvents of random block matrices from RMT, the limiting of the variance is theoretically demonstrated to be increasing for $\psi_2 \in (0, 1)$, decreasing for $\psi_2 \in (1, \infty)$, and diverging as $\psi_2 \to 1$.

In a similar spirit, Mei and Montanari [144] use RMT to study the spectral distribution of the Gram matrix $\boldsymbol{Z} =$

Table 8
Comparison of problem settings on analysis of high dimensional random features on double descent.

studies	metric	data generation				asymptotic?	result
		$\{\boldsymbol{x}_i\}_{i=1}^n$	f_ρ	activation function	$\boldsymbol{W}$		
[143, Theorem 7]	population risk	$\mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$	$\langle \boldsymbol{x}, \boldsymbol{\zeta} \rangle$	normalized	$\mathcal{N}(0, 1/d)$	✓	variance ↗ ↘
[155, Theorem 4]	population risk	$\mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$	$\langle \boldsymbol{x}, \boldsymbol{\zeta} \rangle$	bounded	$\mathcal{N}(0, 1/d)$	✓	variance ↗ ↘
[144, Theorem 2]	expected excess risk	$\mathbb{S}^{d-1}(\sqrt{d})$	$\langle \boldsymbol{x}, \boldsymbol{\zeta} \rangle$ + nonlinear [1]	bounded	Unif($\mathbb{S}^{d-1}(\sqrt{d})$)	✓	variance, bias ↗ ↘
[156]	expected excess risk	$\mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$	$\langle \boldsymbol{x}, \boldsymbol{\zeta} \rangle$	ReLU	$\mathcal{N}(0, 1)$	✓	refined [2]
[157]	generalization error	$\mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$	$f(\langle \boldsymbol{x}, \boldsymbol{\zeta} \rangle)$	general	general	✓	↗ ↘
[158, Theorem 1]	generalization error	$\mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$	$\langle \boldsymbol{x}, \boldsymbol{\zeta} \rangle$	normalized	$\mathcal{N}(0, 1)$	✓	refined [2]
[159, Theorem 1]	generalization error	$\mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$	$\langle \boldsymbol{x}, \boldsymbol{\zeta} \rangle$	general	general	✓	↗ ↘
[160, Proposition 1]	generalization error	$\mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$	$\langle \boldsymbol{x}, \boldsymbol{\zeta} \rangle$	odd, bounded	sub-Gaussian	✓	↗ ↘
[161, Theorem 5.1]	expected excess risk	Gaussian	general	$[\cos(\cdot), \sin(\cdot)]$	$\mathcal{N}(0, 1)$	✗	↗ ↘
[154, Theorem 3]	generalization error	general	- [3]	$[\cos(\cdot), \sin(\cdot)]$	$\mathcal{N}(0, 1)$	✓	↗ ↘

[1] The nonlinear component is a centered isotropic Gaussian process indexed by $\boldsymbol{x} \in \mathbb{S}^{d-1}(\sqrt{d})$.
[2] A refined decomposition on variance is conducted by sources of randomness: "noise variance", "initialization variance", and "sampling variance" to possess each term [156] or their interpretations [158].
[3] It makes no assumptions on f_ρ but requires that test data "behave" statistically like the training data by concentrated random vectors.

$\sigma(\boldsymbol{X}\boldsymbol{W}^\top/\sqrt{d})/\sqrt{d}$ by considering the Stieltjes transform of a related random block matrix, and show that, under least squares regression setting in an asymptotic viewpoint, both the bias and variance have a peak at the interpolation threshold $\psi_2 = 1$ and diverge there when $\lambda \to 0$. Under this framework, according to the randomness stemming from label noise, initialization, and training features, a refined bias-variance decomposition is conducted by [156], [163] and further improved by [158], [164] using the *analysis of variance*. Apart from refined error decomposition schemes, the authors of [155], [157], [159] consider a general setting on convex loss functions, transformation matrix, and activation functions for regression and classification. Here the techniques used for analysis are not limited to RMT. Instead, replica method [165] (a non-rigorous heuristic method from statistical physics) used in [156], [157], [163] and the convex Gaussian min-max (CGMM) theorem [166] used in [159] are two alternative way to derive the desired results. Note that, CGMM requires the data to be Gaussian, which might restrict the application scope of their results but is still a common-used technical tool for max-margin linear classifier [167], boosting classifiers [168], and adversarial training for linear regression [169] in over-parameterized regimes. Admittedly, the applied replica method in statistical physics is quite different from [144] for tackling inverse random matrices in RMT. However, most of the above methods admit the equivalence between the considered data model and the Gaussian covariate model. That means, problem (3) with Gaussian data can be asymptotically equivalent to

$$\min_{\boldsymbol{\beta} \in \mathbb{R}^s} \frac{1}{n} \sum_{i=1}^n \ell\left(y_i, \boldsymbol{\beta}^\top \left(\mu_0 \mathbf{1}_k + \mu_1 \boldsymbol{W}\boldsymbol{x}_i + \mu_\star \boldsymbol{t}_i\right)\right) + \lambda \|\boldsymbol{\beta}\|_2^2,$$

where $\{\boldsymbol{t}_i\}_{i=1}^n \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$, $\mu_0 = \mathbb{E}[\sigma(t)]$, $\mu_1 = \mathbb{E}[t\sigma(t)]$ and $\mu_\star = \mathbb{E}[\sigma(t)^2] - \mu_0^2 - \mu_1^2$ for a standard Gaussian variable t. This equivalence on generalization error in an asymptotic viewpoint is proved in [160].

Different from the above results in an asymptotic view, Jacot et al. [161] present a non-asymptotic result by taking finite-size Stieltjes transform of generalized Wishart matrix, and further argue that random feature models can be close to KRR with an additional regularization. The used technical tool is related to the "calculus of deterministic equivalents" for random matrices [170]. This technique is also used in [154] to derive the exact asymptotic deterministic equivalent of $\mathbb{E}_{\boldsymbol{W}}[(\boldsymbol{Z}\boldsymbol{Z}^\top + n\lambda \boldsymbol{I})^{-1}]$, which captures the asymptotic behavior on double descent. Note that, this work makes no data assumption to match real-world data, which is different from previous work relying on specific data distribution.

7.2 Discussion on Random Features and DNNs

As mentioned, random features models have been fruitfully used to analyze the double descent phenomenon. However, it is non-trivial to transfer results for these models to practical neural networks, which are typically deep but not too wide. There is still a substantial gap between existing theory based on random features and the modern practice of DNNs in approximation ability. For example, under the spherical data setting, Ghorbani et al. [67] (a more general version in [171] on data distribution) point out that as $n \to \infty$, a random features regression model can only fit the projection of the target function onto the space of degree-ℓ polynomials when $s = \Omega(d^{\ell+1-\delta})$ random features are used for some $\delta > 0$. More importantly, if s, d are taken as large with $s = \Omega(d)$, then the function space by random features can only capture linear functions. Even if we consider the NTK model, it can just capture quadratic functions. That means, both random features and NTK have limited approximation power in the lazy training scheme [65]. In addition, Yehudai and Shamir [172] show that the random features model cannot efficiently approximate a single ReLU neuron as it requires the number of random features to be exponentially large in the feature dimension d. This is consistent with the classical result for kernel approximation in the under-parameterized regime: the random features model, QMC, and quadrature based methods require $s = \Omega(\exp(d))$ to achieve an ϵ approximation error [26].

Admittedly, the above results may appear pessimistic due to the simple architecture. Nevertheless, random features is still an effective tool, at least the first step, for analyzing and understanding DNNs in certain regimes, and we believe its potential has yet to be fully exploited. Note that the random features model is still a strong and universal approximator [173] in the sense that the RKHSs induced by a broad class of random features are dense in the space of continuous functions. While the aforementioned results

show that the number of required features may be exponential in the worst case, a more refined analysis can still provide useful insights for DNNs. One potential way forward in deep learning theory is to use the random features model to analyze DNNs *with pruning*. For example, the best paper [174] in the *Seventh International Conference on Learning Representations* (ICLR2019) put forward the following *Lottery Ticket Hypothesis*: a deep neural network with random initialization contains a small sub-network which, when trained in isolation, can compete with the performance of the original one. Malach et al. [54] provide a stronger claim that a randomly-initialized and sufficiently over-parameterized neural network contains a sub-network with nearly the same accuracy as the original one, without any further training. Their analysis points to the equivalence between random features and the sub-network model. As such, the random features model is potentially useful for network pruning [175] in terms of, e.g., guiding the design of neurons pruning for accelerating computations, and understanding network pruning and the full DNNs.

8 Conclusion

In this survey, we systematically review random features based algorithms and their associated theoretical results. We also give an overview on generalization properties of high dimensional random features in over-parameterized regimes on double descent, and discuss the limitations and potential of random features in the theory development for neural networks. Below we provide additional remarks and discuss several open problems that are of interest for future research.

- As a typical data independent method, random features are simpler to implement, easy to parallelize, and naturally apply to streaming or dynamic data. Current efforts on Nyström approximation by a preconditioned gradient solver parallelized with multiple GPUs [176] and quantum algorithms [112] can guide us to design powerful implementation for random features to handling millions/billions data.
- Experimental comparisons show that better kernel approximation does not directly translate to lower generalization errors. We partly answer this question in the current survey but it may be not sufficient to explain this phenomenon. We believe this issue deserves further in-depth study.
- Kernel learning via the spectral density is a popular direction [87], [89], which can be naturally combined with Generative Adversarial Networks (GANs); see [84] for details. In this setting, one may associate the learned model with an implicit probability density that is flexible to characterize the relationships and similarities in the data. This is an interesting area for further research.
- The double descent phenomenon has been observed and studied in random features model by various technical tools under different settings. Current theoretical results, such as those in [144], [154], may be extended to a more general setting with less restricted assumptions on data generation, model formulation, and the target function. Besides, more refined analysis and delicate phenomena beyond double descent have been investigated on the linear model, e.g., multiple descent phenomena [177] and optimal (negative) regularization [178], [179]. Understanding these more delicate phenomena for random features requires further investigation and refined analysis.
- There exist significant gaps between the random features model and practical neural networks, both in theory and empirically. Even for fitting simple quadratic or mixture models, the random features model cannot achieve a zero error with a finite number of neurons in general, while NTK and fully trained networks can [180]. Numerical experiments indicate that the prediction performance of NTK and CNTK may significantly degrade if the random features are generated from practically sized nets [13].
- Despite the limitations of existing theory, random features models are still useful for understanding and improving DNNs. For example, understanding the equivalence between the random features model and weight pruning in the Lottery Ticket Hypothesis [54], may be promising future directions.

We hope that this survey will stimulate further research on the above open problems.

Acknowledgements

The research leading to these results has received funding from the European Research Council under the European Union's Horizon 2020 research and innovation program / ERC Advanced Grant E-DUALITY (787960). This paper reflects only the authors' views and the Union is not liable for any use that may be made of the contained information. This work was supported in part by Research Council KU Leuven: Optimization frameworks for deep kernel machines C14/18/068; Flemish Government: FWO projects: GOA4917N (Deep Restricted Kernel Machines: Methods and Foundations), PhD/Postdoc grant. This research received funding from the Flemish Government (AI Research Program). This work was supported in part by Ford KU Leuven Research Alliance Project KUL0076 (Stability analysis and performance improvement of deep reinforcement learning algorithms), EU H2020 ICT-48 Network TAILOR (Foundations of Trustworthy AI - Integrating Reasoning, Learning and Optimization), Leuven.AI Institute; and in part by the National Natural Science Foundation of China 61977046, in part by National Science Foundation grants CCF-1657420 and CCF-1704828, and in part by SJTU Global Strategic Partnership Fund (2020 SJTU-CORNELL) and Shanghai Municipal Science and Technology Major Project (2021SHZDZX0102).

APPENDIX A
PROOF OF PROPOSITION 1

Proof. It is clear that an exact KRR estimator is $f_{\boldsymbol{z},\lambda}(\boldsymbol{x}) = k(\boldsymbol{x},\boldsymbol{X})(\boldsymbol{K}+n\lambda\boldsymbol{I})^{-1}\boldsymbol{y}$ and its random features based version is $\tilde{f}_{\boldsymbol{z},\lambda}(\boldsymbol{x}) = \tilde{k}(\boldsymbol{x},\boldsymbol{X})(\widetilde{\boldsymbol{K}}+n\lambda\boldsymbol{I})^{-1}\boldsymbol{y}$, where $\widetilde{\boldsymbol{K}} = \boldsymbol{Z}\boldsymbol{Z}^\top$ with $\boldsymbol{Z}\in\mathbb{R}^{n\times s}$. The definition of the excess risk for least squares implies

$$\mathcal{E}(\tilde{f}_{\boldsymbol{z},\lambda}) - \mathcal{E}(f_\rho) = \Big[\mathcal{E}(\tilde{f}_{\boldsymbol{z},\lambda}) - \mathcal{E}(f_{\boldsymbol{z},\lambda})\Big] + [\mathcal{E}(f_{\boldsymbol{z},\lambda}) - \mathcal{E}(f_\rho)] = \|\tilde{f}_{\boldsymbol{z},\lambda} - f_{\boldsymbol{z},\lambda}\|^2 + \|f_{\boldsymbol{z},\lambda} - f_\rho\|^2,$$

where the first term in the right hand is the expected error difference between the original KRR and its random features approximation version. The second term in the right hand is the excess risk of KRR, which is independent of the quality of kernel approximation. Specifically, the first term can be further expressed by the representer theorem

$$\|\tilde{f}_{\boldsymbol{z},\lambda} - f_{\boldsymbol{z},\lambda}\|^2 = \mathbb{E}_{\boldsymbol{x}}[\tilde{f}_{\boldsymbol{z},\lambda}(\boldsymbol{x}) - f_{\boldsymbol{z},\lambda}(\boldsymbol{x})]^2 = \mathbb{E}_{\boldsymbol{x}}\left(\sum_{i=1}^n\Big[\tilde{\alpha}_i\tilde{k}(\boldsymbol{x}_i,\boldsymbol{x}) - \alpha_i k(\boldsymbol{x}_i,\boldsymbol{x})\Big]\right)^2. \tag{29}$$

Intuitively speaking, kernel approximation aims to preserve the inner product in two Hilbert spaces, i.e., $\langle k(\boldsymbol{x},\cdot), k(\boldsymbol{x}',\cdot)\rangle_{\mathcal{H}} \approx \langle\tilde{k}(\boldsymbol{x},\cdot),\tilde{k}(\boldsymbol{x}',\cdot)\rangle_{\widetilde{\mathcal{H}}}$. Nevertheless, the preservation of the inner-product does not immediately guarantee a small value of $\tilde{\alpha}_i\langle\tilde{k}(\boldsymbol{x},\cdot),\tilde{k}(\boldsymbol{x}',\cdot)\rangle_{\widetilde{\mathcal{H}}} - \alpha_i\langle k(\boldsymbol{x},\cdot),k(\boldsymbol{x}',\cdot)\rangle_{\mathcal{H}}$ in Eq. (29).

Informally, the proof idea is the following: define $\widetilde{\boldsymbol{K}} = \boldsymbol{K}+\boldsymbol{E}$ and $\tilde{k}(\boldsymbol{x},\boldsymbol{X}) = k(\boldsymbol{x},\boldsymbol{X}) + \tilde{\boldsymbol{\epsilon}}$ with the residual error matrix $\boldsymbol{E}\in\mathbb{R}^{n\times n}$ and the residual error vector $\tilde{\boldsymbol{\epsilon}}\in\mathbb{R}^{1\times n}$ such that $\tilde{k}(\boldsymbol{x},\boldsymbol{X})\in\mathbb{R}^{1\times n}$. Generally, the residual error $\boldsymbol{E}$ and $\tilde{\boldsymbol{\epsilon}}$ show the consistency, that is, a small kernel approximation error $\|\boldsymbol{E}\|$ implies a small $\|\tilde{\boldsymbol{\epsilon}}\|$. Consider two random features based algorithms A1 and A2 yielding two approximated kernel matrices $\widetilde{\boldsymbol{K}}_1$ and $\widetilde{\boldsymbol{K}}_2$, and their respective KRR estimators $\tilde{f}^{(\mathrm{A1})}_{\boldsymbol{z},\lambda}$ and $\tilde{f}^{(\mathrm{A2})}_{\boldsymbol{z},\lambda}$. The corresponding residual error matrices/vectors are defined as $(\boldsymbol{E}_1,\tilde{\boldsymbol{\epsilon}}_1)$ and $(\boldsymbol{E}_2,\tilde{\boldsymbol{\epsilon}}_2)$ such that $\widetilde{\boldsymbol{K}}_1 := \boldsymbol{K}+\boldsymbol{E}_1$ and $\widetilde{\boldsymbol{K}}_2 := \boldsymbol{K}+\boldsymbol{E}_2$. Without loss of generality, we assume $\|\boldsymbol{E}_1\|\le\|\boldsymbol{E}_2\|$ and $\|\tilde{\boldsymbol{\epsilon}}_1\|\le\|\tilde{\boldsymbol{\epsilon}}_2\|$. In this case, our target is to prove that, there exists one case such that $|\tilde{f}^{(\mathrm{A1})}_{\boldsymbol{z},\lambda}(\boldsymbol{x}) - f_{\boldsymbol{z},\lambda}(\boldsymbol{x})| \ge |\tilde{f}^{(\mathrm{A2})}_{\boldsymbol{z},\lambda}(\boldsymbol{x}) - f_{\boldsymbol{z},\lambda}(\boldsymbol{x})|$. For notational simplicity, denote $T(\boldsymbol{E},\tilde{\boldsymbol{\epsilon}}) := \langle\boldsymbol{y}^\top, k(\boldsymbol{x},\boldsymbol{X})(\boldsymbol{K}+n\lambda\boldsymbol{I})^{-1}\boldsymbol{E} - \tilde{\boldsymbol{\epsilon}}\rangle$, $T_1(\boldsymbol{E}_1,\tilde{\boldsymbol{\epsilon}}_1) := \langle\boldsymbol{y}^\top, k(\boldsymbol{x},\boldsymbol{X})(\boldsymbol{K}+n\lambda\boldsymbol{I})^{-1}\boldsymbol{E}_1 - \tilde{\boldsymbol{\epsilon}}_1\rangle$, and $T_2(\boldsymbol{E}_2,\tilde{\boldsymbol{\epsilon}}_2) := \langle\boldsymbol{y}^\top, k(\boldsymbol{x},\boldsymbol{X})(\boldsymbol{K}+n\lambda\boldsymbol{I})^{-1}\boldsymbol{E}_2 - \tilde{\boldsymbol{\epsilon}}_2\rangle$. To prove our result, we make the following three assumptions:

- I. the residual matrix $\boldsymbol{E}$ is semi-positive definite and $\widetilde{\boldsymbol{K}}_1$, $\widetilde{\boldsymbol{K}}_2$ are non-singular.
- II. $n\lambda\le\lambda_1(\widetilde{\boldsymbol{K}}_1)\le\lambda_1(\widetilde{\boldsymbol{K}}_2)$, and $\widetilde{\boldsymbol{K}}_2$ admits (at least) polynomial decay.
- III. the inner product $\langle\boldsymbol{y}^\top, k(\boldsymbol{x},\boldsymbol{X})(\boldsymbol{K}+n\lambda\boldsymbol{I})^{-1}\boldsymbol{E} - \tilde{\boldsymbol{\epsilon}}\rangle =: T(\boldsymbol{E},\tilde{\boldsymbol{\epsilon}})$ is non-negative.

The above three assumptions are mild, common-used and attainable, see in [31]. Specifically, we only need to prove the existence of our claim: there exists one case such that $|\tilde{f}^{(\mathrm{A1})}_{\boldsymbol{z},\lambda}(\boldsymbol{x}) - f_{\boldsymbol{z},\lambda}(\boldsymbol{x})| \ge |\tilde{f}^{(\mathrm{A2})}_{\boldsymbol{z},\lambda}(\boldsymbol{x}) - f_{\boldsymbol{z},\lambda}(\boldsymbol{x})|$ under $\|\boldsymbol{E}_1\|\le\|\boldsymbol{E}_2\|$ and $\|\tilde{\boldsymbol{\epsilon}}_1\|\le\|\tilde{\boldsymbol{\epsilon}}_2\|$. Therefore, the above assumptions could be further relaxed.

According to Eq. (29), for a new sample $\boldsymbol{x}$, we in turn focus on $|\tilde{f}_{\boldsymbol{z},\lambda}(\boldsymbol{x}) - f_{\boldsymbol{z},\lambda}(\boldsymbol{x})|$, which can be upper bounded by

$$\begin{aligned}
|\tilde{f}_{\boldsymbol{z},\lambda}(\boldsymbol{x}) - f_{\boldsymbol{z},\lambda}(\boldsymbol{x})| &= |k(\boldsymbol{x},\boldsymbol{X})(\boldsymbol{K}+n\lambda\boldsymbol{I})^{-1}\boldsymbol{y} - [k(\boldsymbol{x},\boldsymbol{X})+\tilde{\boldsymbol{\epsilon}}](\boldsymbol{K}+\boldsymbol{E}+n\lambda\boldsymbol{I})^{-1}\boldsymbol{y}| \\
&= |k(\boldsymbol{x},\boldsymbol{X})[(\boldsymbol{K}+n\lambda\boldsymbol{I})^{-1} - (\boldsymbol{K}+n\lambda\boldsymbol{I}+\boldsymbol{E})^{-1}]\boldsymbol{y} - \tilde{\boldsymbol{\epsilon}}(\boldsymbol{K}+n\lambda\boldsymbol{I}+\boldsymbol{E})^{-1}\boldsymbol{y}| \\
&= |[k(\boldsymbol{x},\boldsymbol{X})(\boldsymbol{K}+n\lambda\boldsymbol{I})^{-1}\boldsymbol{E} - \tilde{\boldsymbol{\epsilon}}](\boldsymbol{K}+n\lambda\boldsymbol{I}+\boldsymbol{E})^{-1}\boldsymbol{y}| \\
&\le [k(\boldsymbol{x},\boldsymbol{X})(\boldsymbol{K}+n\lambda\boldsymbol{I})^{-1}\boldsymbol{E} - \tilde{\boldsymbol{\epsilon}}]\boldsymbol{y}\sum_{i=1}^n\frac{1}{\lambda_i(\boldsymbol{K}+\boldsymbol{E})+n\lambda} \\
&= \langle\boldsymbol{y}^\top, (k(\boldsymbol{x},\boldsymbol{X})(\boldsymbol{K}+n\lambda\boldsymbol{I})^{-1}\boldsymbol{E} - \tilde{\boldsymbol{\epsilon}}\rangle\sum_{i=1}^n\frac{1}{\lambda_i(\boldsymbol{K}+\boldsymbol{E})+n\lambda} =: \sum_{i=1}^n\frac{T(\boldsymbol{E},\tilde{\boldsymbol{\epsilon}})}{\lambda_i(\boldsymbol{K}+\boldsymbol{E})+n\lambda}
\end{aligned} \tag{30}$$

where the third equality holds by $\boldsymbol{A}^{-1}-\boldsymbol{B}^{-1} = \boldsymbol{A}^{-1}(\boldsymbol{B}-\boldsymbol{A})\boldsymbol{B}^{-1}$. The first inequality derives from $\boldsymbol{a}^\top\boldsymbol{A}\boldsymbol{b}\le\boldsymbol{a}^\top\boldsymbol{b}\,\mathrm{tr}(\boldsymbol{A})$ for two semi-positive definite matrix $\boldsymbol{A}$ and $\boldsymbol{b}^\top\boldsymbol{a}$ (which can be derived from the used assumptions). Further, by virtue of $\boldsymbol{a}^\top\boldsymbol{A}\boldsymbol{b}\ge\lambda_n(\boldsymbol{A})\,\mathrm{tr}(\boldsymbol{a}^\top\boldsymbol{b})$, the error $|\tilde{f}_{\boldsymbol{z},\lambda}(\boldsymbol{x}) - f_{\boldsymbol{z},\lambda}(\boldsymbol{x})|$ can be lower bounded by

$$\begin{aligned}
|\tilde{f}_{\boldsymbol{z},\lambda}(\boldsymbol{x}) - f_{\boldsymbol{z},\lambda}(\boldsymbol{x})| &= |[k(\boldsymbol{x},\boldsymbol{X})(\boldsymbol{K}+n\lambda\boldsymbol{I})^{-1}\boldsymbol{E} - \tilde{\boldsymbol{\epsilon}}](\boldsymbol{K}+n\lambda\boldsymbol{I}+\boldsymbol{E})^{-1}\boldsymbol{y}| \\
&\ge \frac{\langle\boldsymbol{y}^\top, [k(\boldsymbol{x},\boldsymbol{X})(\boldsymbol{K}+n\lambda\boldsymbol{I})^{-1}\boldsymbol{E} - \tilde{\boldsymbol{\epsilon}}]\rangle}{\lambda_1(\boldsymbol{K}+\boldsymbol{E})+n\lambda} =: \frac{T(\boldsymbol{E},\tilde{\boldsymbol{\epsilon}})}{\lambda_1(\boldsymbol{K}+\boldsymbol{E})+n\lambda}
\end{aligned} \tag{31}$$

Combining Eqs. (30) and (31), we have

$$0\le\frac{T(\boldsymbol{E},\tilde{\boldsymbol{\epsilon}})}{\lambda_1(\boldsymbol{K}+\boldsymbol{E})+n\lambda}\le|\tilde{f}_{\boldsymbol{z},\lambda}(\boldsymbol{x}) - f_{\boldsymbol{z},\lambda}(\boldsymbol{x})|\le\sum_{i=1}^n\frac{T(\boldsymbol{E},\tilde{\boldsymbol{\epsilon}})}{\lambda_i(\boldsymbol{K}+\boldsymbol{E})+n\lambda}. \tag{32}$$

Considering such two algorithms A1 and A2, under the condition of $\|\boldsymbol{E}_1\|\le\|\boldsymbol{E}_2\|$ and $\|\tilde{\boldsymbol{\epsilon}}_1\|\le\|\tilde{\boldsymbol{\epsilon}}_2\|$, there exists one case such that $T_1(\boldsymbol{E}_1,\tilde{\boldsymbol{\epsilon}}_1)\ge T_2(\boldsymbol{E}_2,\tilde{\boldsymbol{\epsilon}}_2)$, i.e.,

$$\langle\boldsymbol{y}^\top, (k(\boldsymbol{x},\boldsymbol{X})(\boldsymbol{K}+n\lambda\boldsymbol{I})^{-1}\boldsymbol{E}_1 - \tilde{\boldsymbol{\epsilon}}_1\rangle\ge\langle\boldsymbol{y}^\top, (k(\boldsymbol{x},\boldsymbol{X})(\boldsymbol{K}+n\lambda\boldsymbol{I})^{-1}\boldsymbol{E}_2 - \tilde{\boldsymbol{\epsilon}}_2\rangle, \tag{33}$$

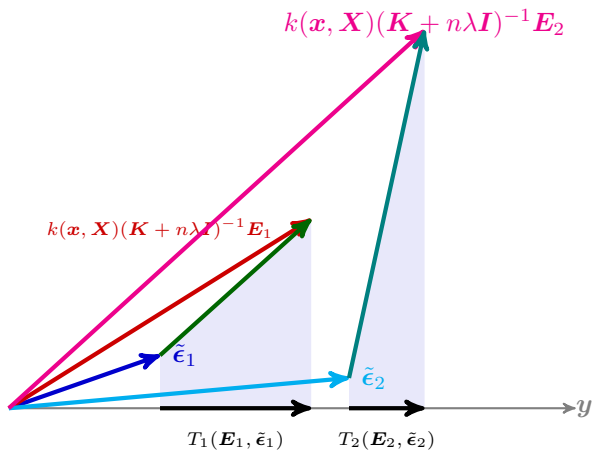


Figure 8. Illustration of the geometric proof for one case such that $T_1(\boldsymbol{E}_1,\tilde{\boldsymbol{\epsilon}}_1) \geq cT_2(\boldsymbol{E}_2,\tilde{\boldsymbol{\epsilon}}_2)$ under the condition of $\|\boldsymbol{E}_1\| \leq \|\boldsymbol{E}_2\|$ and $\|\tilde{\boldsymbol{\epsilon}}_1\| \leq \|\tilde{\boldsymbol{\epsilon}}_2\|$, where c is some constant.

which can be achieved by a geometry explanation in Figure 8. By virtue of Eq. (33) and Assumption II, we have

$$\frac{T_1(\boldsymbol{E}_1,\tilde{\boldsymbol{\epsilon}}_1)}{\lambda_1(\boldsymbol{K}+\boldsymbol{E}_1)+n\lambda} - \frac{T_2(\boldsymbol{E}_2,\tilde{\boldsymbol{\epsilon}}_2)}{\lambda_1(\boldsymbol{K}+\boldsymbol{E}_2)+n\lambda} =: \widetilde{C} \geq 0\,.$$

The above inequality implies

$$\widetilde{C} - \sum_{i=2}^{n} \frac{T_2(\boldsymbol{E}_2,\tilde{\boldsymbol{\epsilon}}_2)}{\lambda_i(\boldsymbol{K}+\boldsymbol{E}_2)+n\lambda} \leq \left|\tilde{f}^{(\mathrm{A1})}_{\boldsymbol{z},\lambda}(\boldsymbol{x}) - f_{\boldsymbol{z},\lambda}(\boldsymbol{x})\right| - \left|\tilde{f}^{(\mathrm{A2})}_{\boldsymbol{z},\lambda}(\boldsymbol{x}) - f_{\boldsymbol{z},\lambda}(\boldsymbol{x})\right| \leq \widetilde{C} + \sum_{i=2}^{n} \frac{T_1(\boldsymbol{E}_1,\tilde{\boldsymbol{\epsilon}}_1)}{\lambda_i(\boldsymbol{K}+\boldsymbol{E}_1)+n\lambda}\,.$$

The left-hand of the above inequality can be further improved as

$$\begin{aligned}
\left|\tilde{f}^{(\mathrm{A1})}_{\boldsymbol{z},\lambda}(\boldsymbol{x}) - f_{\boldsymbol{z},\lambda}(\boldsymbol{x})\right| - \left|\tilde{f}^{(\mathrm{A2})}_{\boldsymbol{z},\lambda}(\boldsymbol{x}) - f_{\boldsymbol{z},\lambda}(\boldsymbol{x})\right| &\geq \widetilde{C} - \sum_{i=2}^{n} \frac{T_2(\boldsymbol{E}_2,\tilde{\boldsymbol{\epsilon}}_2)}{\lambda_i(\boldsymbol{K}+\boldsymbol{E}_2)+n\lambda} \\
&\geq \frac{T_1(\boldsymbol{E}_1,\tilde{\boldsymbol{\epsilon}}_1)}{\lambda_1(\boldsymbol{K}+\boldsymbol{E}_2)+n\lambda} - \sum_{i=1}^{n} \frac{T_2(\boldsymbol{E}_2,\tilde{\boldsymbol{\epsilon}}_2)}{\lambda_i(\boldsymbol{K}+\boldsymbol{E}_2)+n\lambda} \\
&\geq \frac{T_1(\boldsymbol{E}_1,\tilde{\boldsymbol{\epsilon}}_1)}{\lambda_1(\boldsymbol{K}+\boldsymbol{E}_2)+n\lambda} - \frac{T_2(\boldsymbol{E}_2,\tilde{\boldsymbol{\epsilon}}_2)}{\lambda_n(\boldsymbol{K}+\boldsymbol{E}_2)} \sum_{i=1}^{n} \frac{\lambda_i(\boldsymbol{K}+\boldsymbol{E}_2)}{\lambda_i(\boldsymbol{K}+\boldsymbol{E}_2)+n\lambda} \\
&\geq \frac{T_1(\boldsymbol{E}_1,\tilde{\boldsymbol{\epsilon}}_1)}{2\lambda_1(\widetilde{\boldsymbol{K}}_2)} - \frac{T_2(\boldsymbol{E}_2,\tilde{\boldsymbol{\epsilon}}_2)}{\lambda_n(\widetilde{\boldsymbol{K}}_2)} d^{\lambda}_{\widetilde{\boldsymbol{K}}_2}\,,
\end{aligned}$$

where $d^{\lambda}_{\widetilde{\boldsymbol{K}}_2}$ is the "effective dimension" of $\widetilde{\boldsymbol{K}}_2$ defined in Eq. (21) and the last inequality follows from Assumption II.

According to the above result, $\left|\tilde{f}^{(\mathrm{A1})}_{\boldsymbol{z},\lambda}(\boldsymbol{x}) - f_{\boldsymbol{z},\lambda}(\boldsymbol{x})\right| - \left|\tilde{f}^{(\mathrm{A2})}_{\boldsymbol{z},\lambda}(\boldsymbol{x}) - f_{\boldsymbol{z},\lambda}(\boldsymbol{x})\right| \geq 0$ holds by the following condition

$$T_1(\boldsymbol{E}_1,\tilde{\boldsymbol{\epsilon}}_1) \geq 2 \underbrace{\left[\frac{\lambda_1(\widetilde{\boldsymbol{K}}_2)}{\lambda_n(\widetilde{\boldsymbol{K}}_2)}\right] d^{\lambda}_{\widetilde{\boldsymbol{K}}_2}}_{=\mathcal{O}(1)} T_2(\boldsymbol{E}_2,\tilde{\boldsymbol{\epsilon}}_2)\,.$$

We observe that, an invertible matrix $\widetilde{\boldsymbol{K}}_2$ admits a finite condition number $\lambda_1(\widetilde{\boldsymbol{K}}_2)/\lambda_n(\widetilde{\boldsymbol{K}}_2) < \infty$. Besides, a fast polynomial eigenvalue decay of $\widetilde{\boldsymbol{K}}_2$ ensures the effective dimension $d^{\lambda}_{\widetilde{\boldsymbol{K}}_2}$ to be finite, which can be obtained by Assumption 5 with $\gamma = 0$. Accordingly, in this case, when these condition are satisfied, $T_1(\boldsymbol{E}_1,\tilde{\boldsymbol{\epsilon}}_1) \geq cT_2(\boldsymbol{E}_2,\tilde{\boldsymbol{\epsilon}}_2)$ can be achieved for some constant c, which can be intuitively observed by Figure 8. Finally, we conclude the proof for existence.

□

APPENDIX B
EXPERIMENTS

In this section, we detail the experimental settings and present the comparison results on the compared approaches on several benchmark datasets across various kernels. This part is organized as follows.

- In Section B.1, we present experimental results across the Gaussian kernel on eight non-image datasets in terms of approximation error, the time cost (sec.) for generating random features mappings, classification accuracy by linear regression and liblinear.

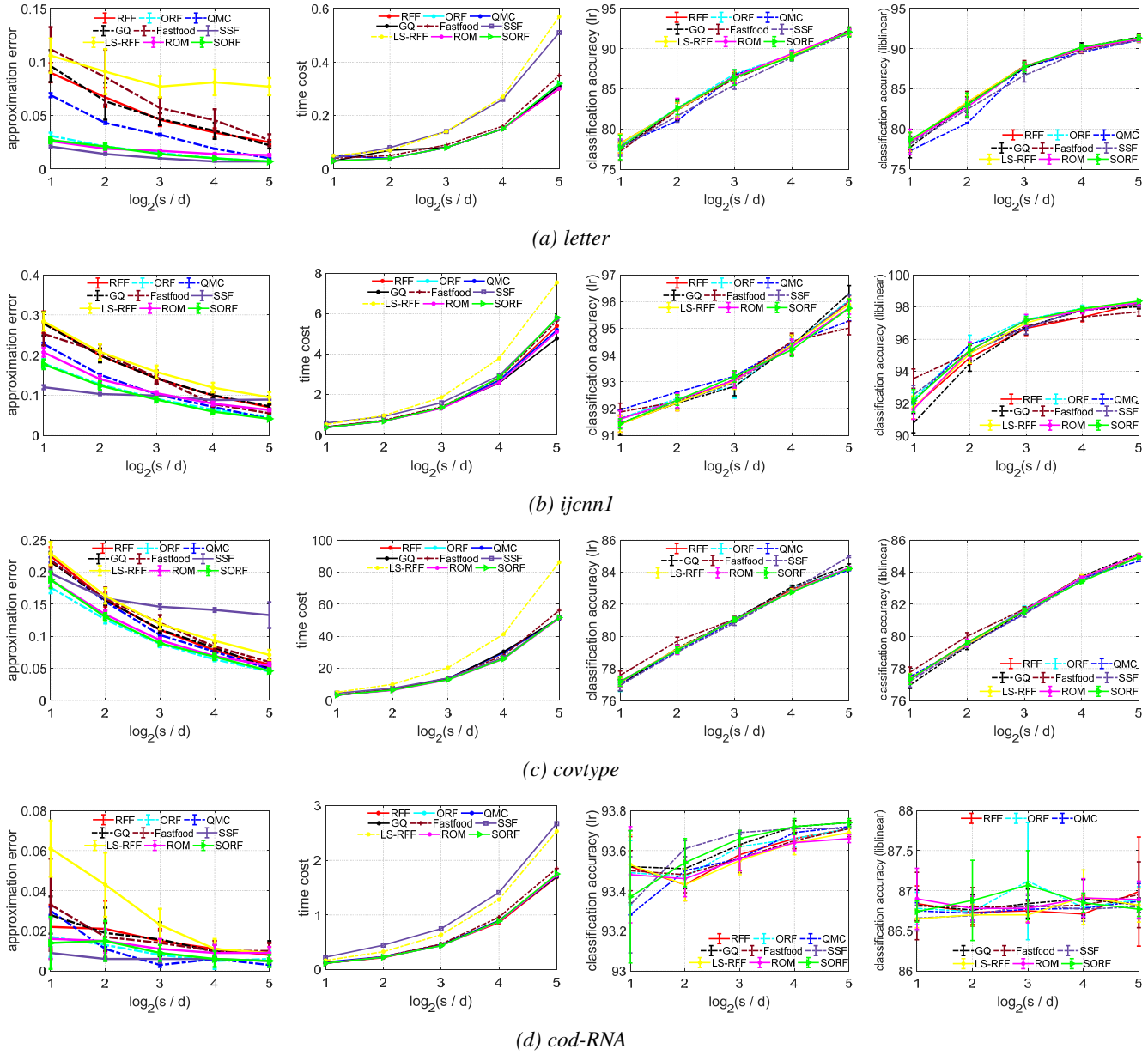


Figure 9. Results of various algorithms across the Gaussian kernel on the *letter*, *ijcnn1*, *covtype*, *cod-RNA* datasets.

- Results on approximation error and test accuracy (by linear regression) across arc-cosine kernels and polynomial kernels are presented in Sections B.2 and B.3, respectively.
- In Section B.4, a ultra-large scale dataset is applied to further validate the related algorithms.

B.1 Results on Gaussian kernels

Figures 9, 10 show the approximation error for the Gaussian kernel, the time cost (sec.) of generating randomized feature mappings, and the test accuracy yielded by linear regression and liblinear on the eight datasets, respectively. We see that as the number of random features increases, these algorithms achieve a smaller approximation error and a higher classification accuracy for both classifiers. We notice some interesting phenomena in terms of the relation between approximation quality and prediction performance, depending on whether the feature dimension is low (i.e., $s = 2d$ or $s = 4d$) or high (i.e., $s = 16d$ or $s = 32d$). In particular, the algorithms with the best kernel approximation performance are often different in the low-dimensional case and the high dimensional case. Therefore, no algorithm always dominate the others. On the other hand, while the approximation quality of these algorithms varies, their prediction performance are often similar. Further, to better understand the above observations, we summarize the best performing algorithm on each dataset in terms of the approximation quality and classification accuracy in Table 7, as illustrated in our main text (refer to Section 6.2.1).

Regarding to computational efficiency, most algorithms achieve the similar time cost on generating random features except SSF and LS-RFF. SSF requires constructing the transformation matrix by minimizing the discrete Riesz 0-energy in advance; LS-RFF is a

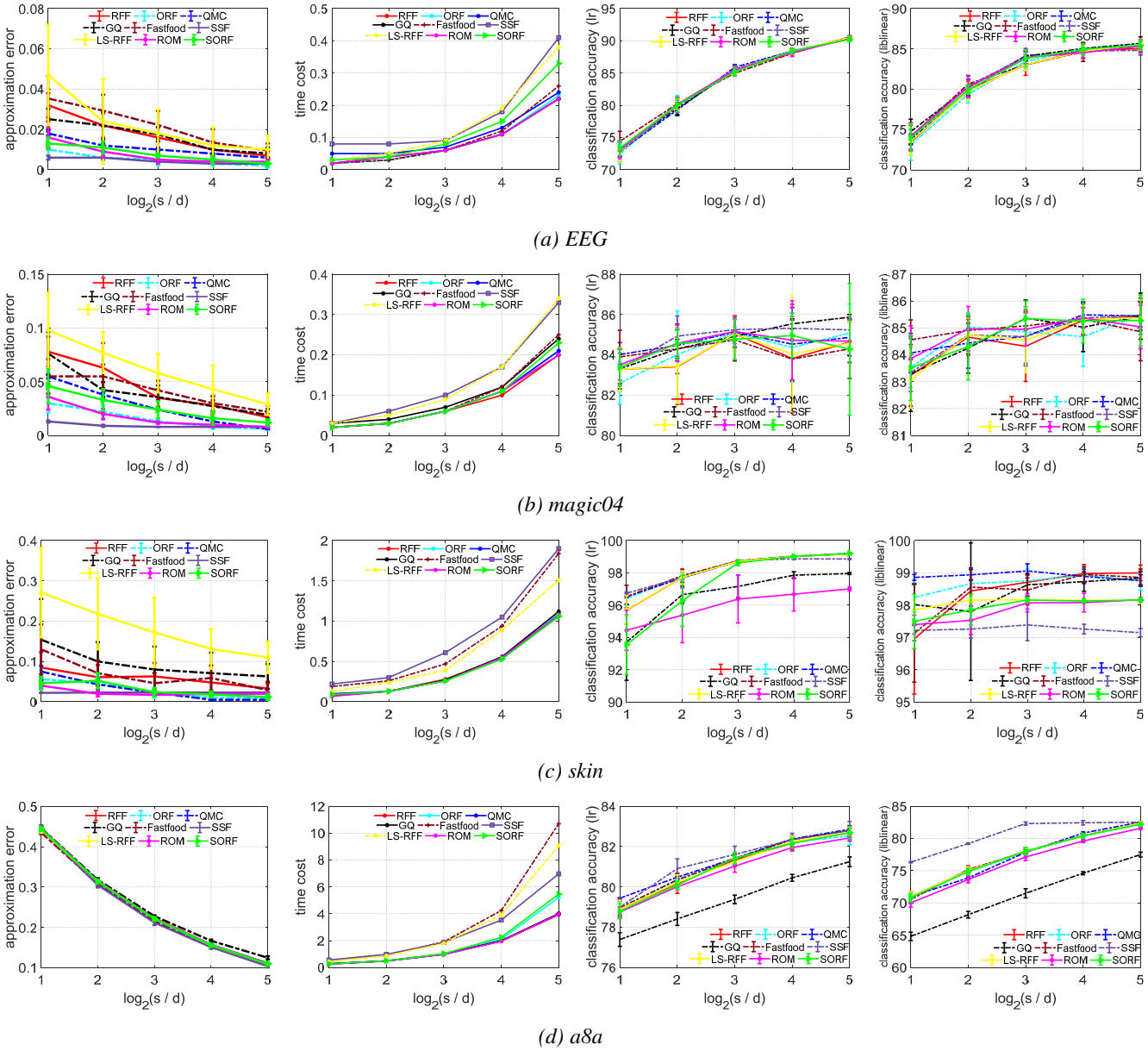


Figure 10. Results of various algorithms across the Gaussian kernel on the *EEG*, *magic04*, *skin*, *a8a* datasets.

data-dependent algorithm that needs to calculate the approximated ridge leverage score. Nevertheless, Fastfood/SORF/ROM does not achieve the reduction on time cost, which appears contradictory to the underlying theoretical result on time complexity. This might be because, one hand, the feature dimension of the used datasets in our experiments often ranges from 10 to 100, except for the image datasets. In this case, it appears difficult to observe the computational saving from $\mathcal{O}(sd)$ to $\mathcal{O}(s \log d)$ or $\mathcal{O}(d \log d)$. On the other hand, in our experiments, due to the relatively inefficient Matlab implementation of Fast Discrete Walsh-Hadamard Transform, typical algorithms (e.g., Fastfood/SORF/ROM) do not show a significant reduction on computational efficiency than RFF.

B.2 Results on Arc-cosine kernels

As mentioned before, according to Eq. (6), various algorithms based on different sampling strategies can be still applicable to arc-cosine kernels, e.g., ORF, QMC, and Fastfood. Accordingly, eight representative algorithms are taken into comparison on arc-cosine kernels, including RFF, ORF, SORF, ROM, Fastfood, QMC, SSF, and GQ.

Figures 11, 12 show the approximation error and test accuracy across the zero/first-order arc-cosine kernels, respectively. It can be observed that in most cases SSF and QMC achieve a lower approximation error than the other approaches, which corresponds to the theoretical findings. However, there is no distinct difference on approximation between RFF and ORF/SORF. In fact, the current theoretical results on ORF/SORF for variance reduction are only valid to the Gaussian kernel. Whether such results can be transferred to arc-cosine kernels are still unclear. In general, the approximation performance and time cost (see Figure 14(a) and 14(b)) of these algorithms on arc-cosine kernels are similar to that on the Gaussian kernel, though the approximation error value is often larger than

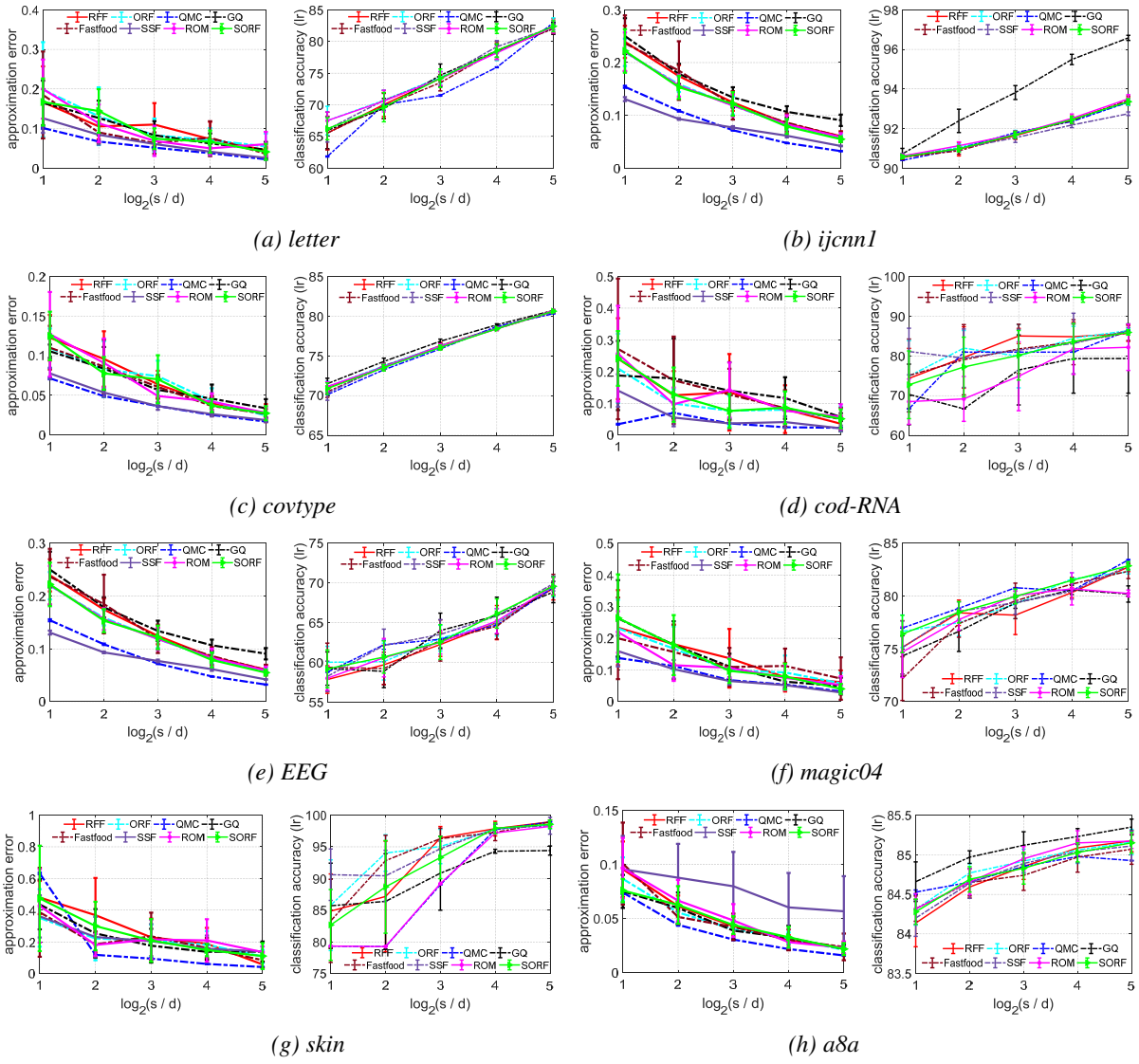


Figure 11. Results on eight datasets across the zero-order arc-cosine kernel.

that for the Gaussian kernel. This is because, according to Eq. (6), we actually conduct a d-dimensional integration approximation, the smoothness of the integrand $\sigma(\boldsymbol{\omega}^\top \boldsymbol{x})\sigma(\boldsymbol{\omega}^\top \boldsymbol{x}')$ would significantly effect the approximation performance as indicated by sampling theory. In the term of classification performance, the difference in test accuracy of most algorithms is relatively small, which shows the similar tendency with that of the Gaussian kernel.

B.3 Results on Polynomial kernels

For polynomial kernel approximation, we include three representative approaches, tensorized random projections (TRP) [74], TensorSketch (TS) [73], and random Maclaurin (RM) [34] sketch evaluated on eight datasets for approximation and prediction. Since the polynomial kernel can be written as a special type of tensor product, TS and TRP work in this setting by sketching a tensor product of arbitrary vectors, which is different from RM using Maclaurin expansion. Figure 13 shows that, TS and TRP have the similar test accuracy, but significantly perform better than RM, as RM's generality is not required for the polynomial kernel. Besides, Figure 14(c) shows that RM is quite computational efficient due to its Maclaurin expansion scheme; while TS takes much time on generating random features since it utilizes a fixed sampling probability to compute the tensor sketch; while TRP works in a flexible sampling strategy proportional to its Maclaurin coefficient.

B.4 Results on the *MNIST-8M* dataset

Here we evaluate the compared ten algorithms across the Gaussian kernel and arc-cosine kernels on the *MNIST-8M* dataset [139]. Due to the memory limit, following the doubly stochastic framework [38], we incorporate these random features based approaches under the data streaming setting for the reduction of time and space complexity. The experimental setting on this dataset follows with [38]: the feature dimension $d = 784$ is reduced to 100 by PCA; the number of random features s is set to 4096; the used Gaussian RBF

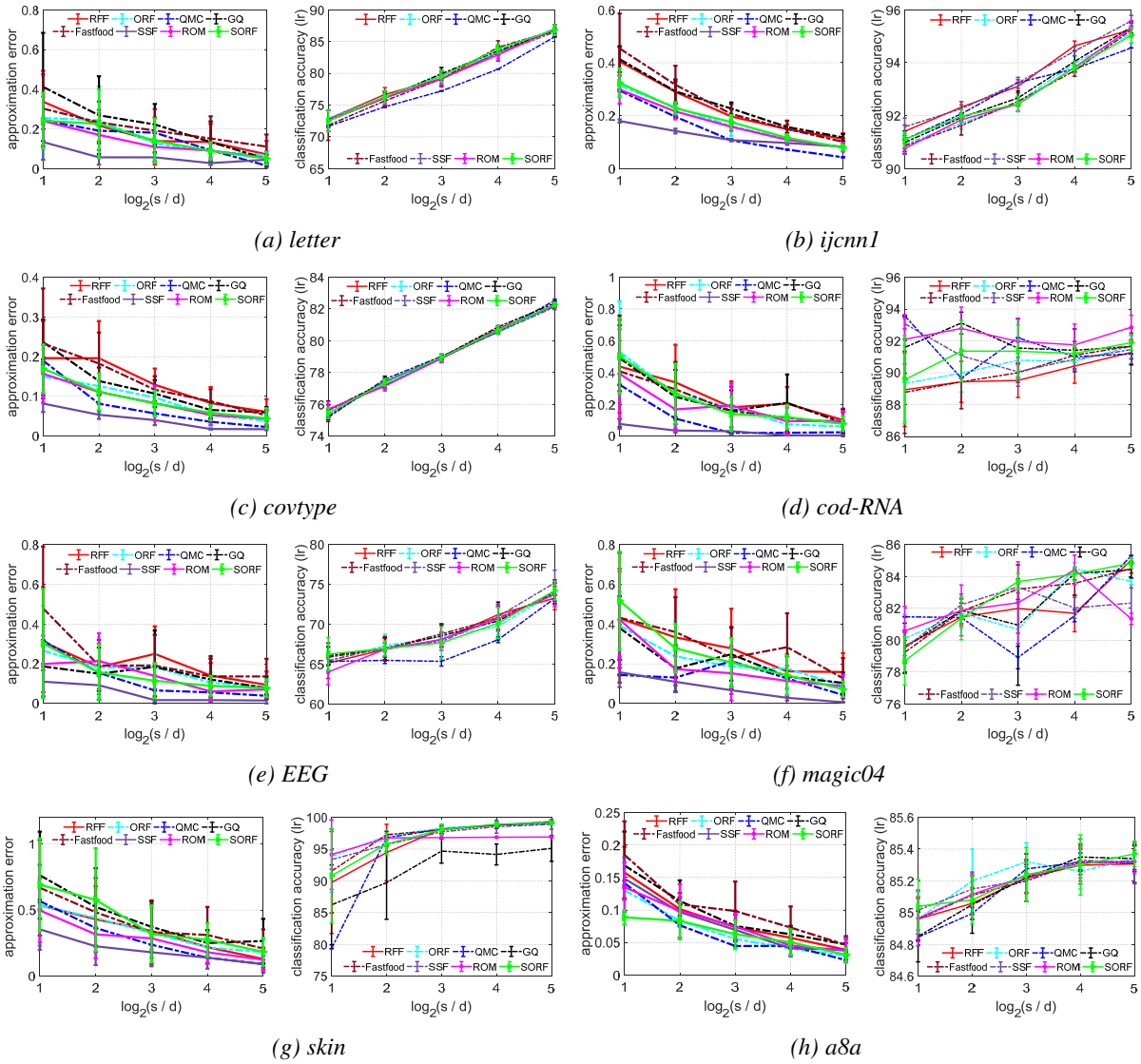


Figure 12. Results on eight datasets across the first-order arc-cosine kernel.

kernel with kernel bandwidth ς equaling to four times the median pairwise distance; logistic regression with the regularization parameter $\lambda = 0.0005$ for this multi-class classification task; the batch size is set to be 2^{20} and feature block to be 2^{15}. Besides, we report the total time cost of each algorithm on generating feature mapping, training process and test process for evaluation.

Table 9 reports the approximation error, training error, test error, and the total time cost of each algorithm across the Gaussian kernel and the zero/first-order arc-cosine kernels under $s = 4096$. It can be found that, ORF/SORF and SSF achieve the best approximation performance on the Gaussian kernel, but ORF fails to significantly improve the approximation ability on arc-cosine kernels. This is consistent with previous discussion on medium datasets in Section B.2.

REFERENCES

[1] Bernhard Schölkopf and Alexander J. Smola, *Learning with kernels: support vector machines, regularization, optimization, and beyond*, MIT Press, 2003.

[2] Johan A.K. Suykens, Tony Van Gestel, Jos De Brabanter, Bart De Moor, and Joos Vandewalle, *Least Squares Support Vector Machines*, World Scientific, 2002.

[3] Mehran Kafai and Kave Eshghi, "CROification: accurate kernel classification with the efficiency of sparse linear SVM," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 1, pp. 34–48, 2019.

[4] Cho-Jui Hsieh, Si Si, and Inderjit Dhillon, "A divide-and-conquer solver for kernel support vector machines," in *International Conference on Machine Learning*, 2014, pp. 566–574.

[5] Yuchen Zhang, John Duchi, and Martin Wainwright, "Divide and conquer kernel ridge regression," in *Conference on Learning Theory*, 2013, pp. 592–617.

[6] Fanghui Liu, Xiaolin Huang, Chen Gong, Jie Yang, and Li Li, "Learning data-adaptive non-parametric kernels," *Journal of Machine Learning Research*, vol. 21, no. 208, pp. 1–39, 2020.

[7] Alex J. Smola and Bernhard Schölkopf, "Sparse greedy matrix approximation for machine learning," in *International Conference on Machine Learning*, 2000, pp. 911–918.

[8] Christopher K.I. Williams and Matthias Seeger, "Using the Nyström method to speed up kernel machines," in *Advances in Neural Information Processing Systems*, 2001, pp. 682–688.

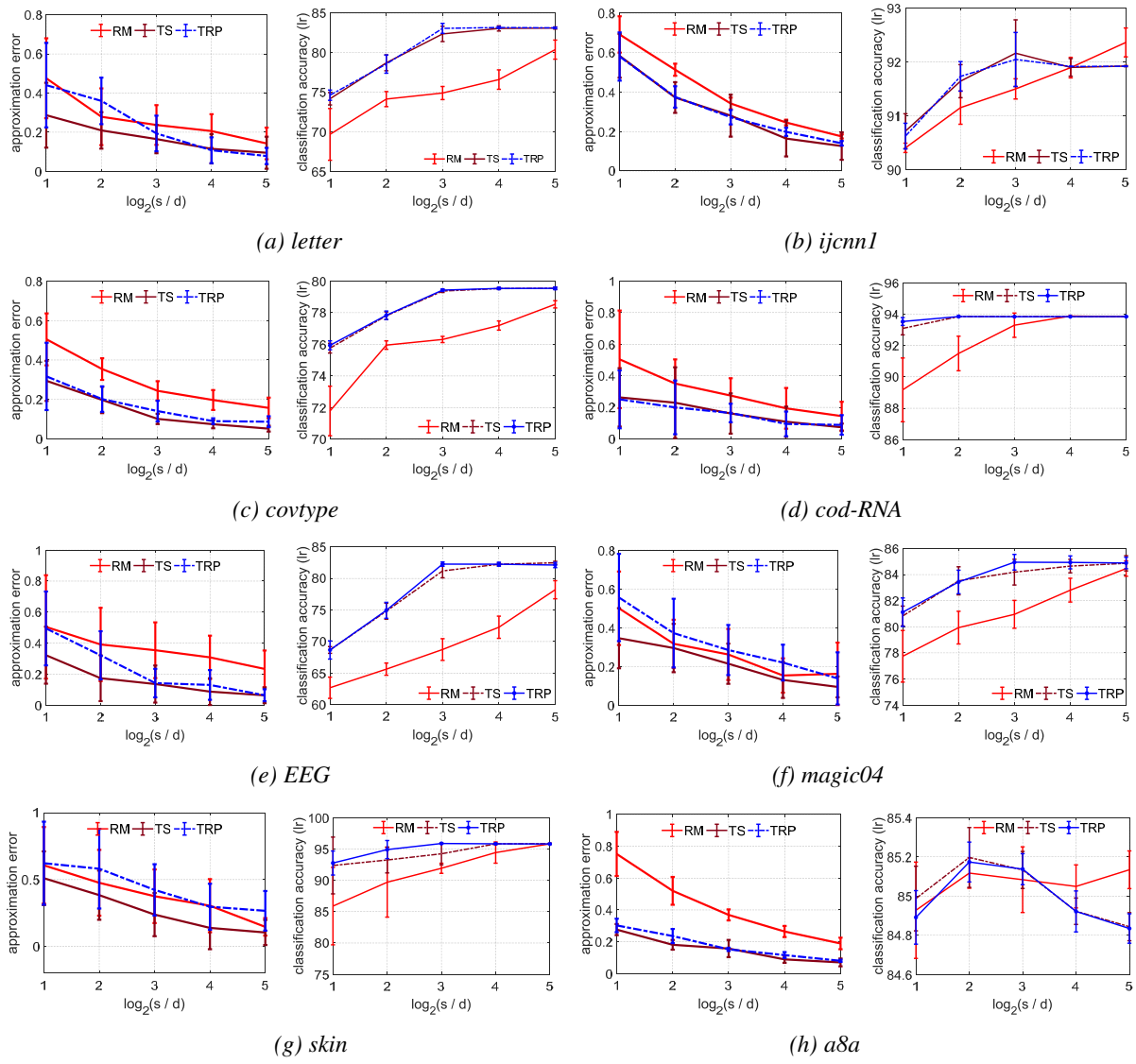


Figure 13. Results on eight datasets across polynomial kernels.

[9] Ali Rahimi and Benjamin Recht, "Random features for large-scale kernel machines," in *Advances in Neural Information Processing Systems*, 2007, pp. 1177–1184.

[10] David Lopez-Paz, Suvrit Sra, Alex J. Smola, Zoubin Ghahramani, and Bernhard Schölkopf, "Randomized nonlinear component analysis," in *International Conference on Machine Learning*, 2014, pp. 1359–1367.

[11] Yitong Sun, Anna Gilbert, and Ambuj Tewari, "But how does it work in theory? Linear SVM with random features," in *Advances in Neural Information Processing Systems*, 2018, pp. 3383–3392.

[12] Arthur Jacot, Franck Gabriel, and Clément Hongler, "Neural tangent kernel: Convergence and generalization in neural networks," in *Advances in Neural Information Processing Systems*, 2018, pp. 8571–8580.

[13] Sanjeev Arora, Simon S. Du, Wei Hu, Zhiyuan Li, Russ R. Salakhutdinov, and Ruosong Wang, "On exact computation with an infinitely wide neural net," in *Advances in Neural Information Processing Systems*, 2019, pp. 8139–8148.

[14] Amir Zandieh, Insu Han, Haim Avron, Neta Shoham, Chaewon Kim, and Jinwoo Shin, "Scaling neural tangent kernels via sketching and random features," *arXiv preprint arXiv:2106.07880*, 2021.

[15] Simon S Du, Kangcheng Hou, Barnabás Póczos, Ruslan Salakhutdinov, Ruosong Wang, and Keyulu Xu, "Graph neural tangent kernel: Fusing graph neural networks with graph kernels," in *Advances in Neural Information Processing Systems*, 2019, pp. 1–11.

[16] Daniele Zambon, Cesare Alippi, and Lorenzo Livi, "Graph random neural features for distance-preserving graph representations," in *International Conference on Machine Learning*. PMLR, 2020, pp. 10968–10977.

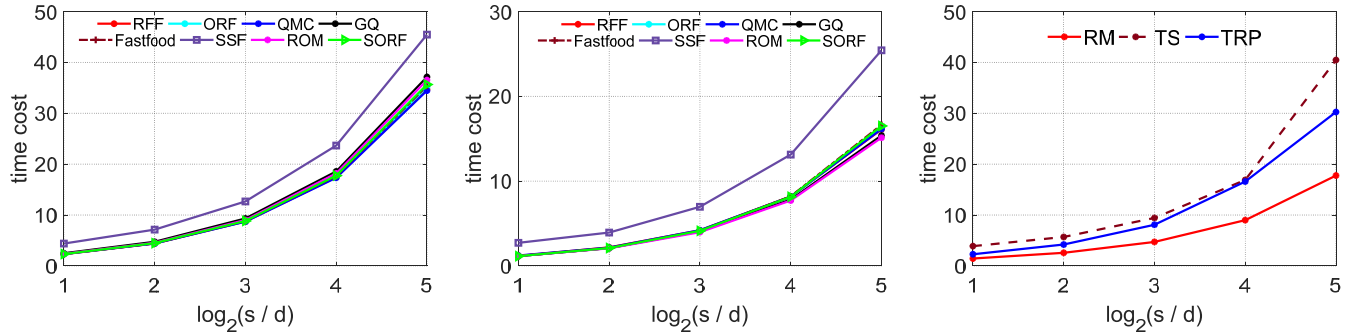
[17] Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, and Weller Adrian, "Rethinking attention with performers," in *International Conference on Learning Representations*, 2021.

[18] Hao Peng, Nikolaos Pappas, Dani Yogatama, Roy Schwartz, Noah Smith, and Lingpeng Kong, "Random feature attention," in *International Conference on Learning Representations*, 2021, pp. 1–19.

[19] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals, "Understanding deep learning requires rethinking generalization," *arXiv preprint arXiv:1611.03530*, 2016.

[20] Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal, "Reconciling modern machine-learning practice and the classical bias–variance trade-off," *the National Academy of Sciences*, vol. 116, no. 32, pp. 15849–15854, 2019.

[21] Yuan Cao and Quanquan Gu, "Generalization bounds of stochastic gradient descent for wide and deep neural networks," in *Advances in Neural Information Processing Systems*, 2019, pp. 10835–10845.



(a) zero-order arc-cosine kernel (b) first-order arc-cosine kernel (c) polynomial kernel

Figure 14. Comparison of various algorithms on the *covtype* dataset in terms of time cost for generating random features mappings.

Table 9
Comparison results of various algorithms across three kernels in terms of training error (%), classification error (%) and total time cost (sec.) on the ultra-large *MNIST 8M* dataset.

kernel	metric	RFF	QMC	ORF	SORF	ROM	Fastfood	SSF	GQ	LS-RFF
Gaussian	approximation error	0.0126	0.0065	0.0041	0.0041	0.0046	0.0159	0.0078	0.0121	0.0147
	training error	0.22%	0.21%	0.19%	0.22%	0.19%	0.21%	0.20%	0.21%	0.22%
	test error	0.99%	1.07%	1.11%	1.13%	0.99%	1.16%	1.09%	1.16%	0.97%
	time cost (sec.)	13669	13999	14296	14526	13497	14343	14872	14322	15725
arccos0	approximation error	0.0209	0.0124	0.0224	0.0231	0.0199	0.0246	0.0448	0.0383	0.0612
	training error	2.71%	2.70%	2.70%	2.70%	2.70%	2.60%	2.70%	3.02%	2.64%
	test error	2.76%	2.91%	2.75%	2.86%	2.73%	2.94%	2.89%	3.00%	2.72%
	time cost (sec.)	10577	10266	10501	10558	10595	10807	11235	10330	12231
arccos1	approximation error	0.0394	0.0104	0.0310	0.0316	0.0259	0.0458	0.0198	0.0369	0.0357
	training error	0.93%	0.96%	0.94%	1.00%	0.94%	0.98%	0.95%	0.96%	0.93%
	test error	1.64%	1.59%	1.52%	1.57%	1.62%	1.27%	1.34%	1.51%	1.62%
	time cost (sec.)	9243.7	9170.3	9187.4	8861.6	8870.8	8824.1	9455.3	9188.1	9742.3

[22] Sanjeev Arora, Simon Du, Wei Hu, Zhiyuan Li, and Ruosong Wang, "Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks," in *International Conference on Machine Learning*, 2019, pp. 322–332.

[23] Ziwei Ji and Matus Telgarsky, "Polylogarithmic width suffices for gradient descent to achieve arbitrarily small test error with shallow ReLU networks," in *International Conference on Learning Representations*, 2020, pp. 1–8.

[24] Felix Xinnan Yu, Ananda Theertha Suresh, Krzysztof Choromanski, Daniel Holtmannrice, and Sanjiv Kumar, "Orthogonal random features," in *Advances in Neural Information Processing Systems*, 2016, pp. 1975–1983.

[25] Haim Avron, Vikas Sindhwani, Jiyan Yang, and Michael W. Mahoney, "Quasi-Monte Carlo feature maps for shift-invariant kernels," *Journal of Machine Learning Research*, vol. 17, no. 1, pp. 4096–4133, 2016.

[26] Tri Dao, Christopher M. De Sa, and Christopher Ré, "Gaussian quadrature for kernel features," in *Advances in neural information processing systems*, 2017, pp. 6107–6117.

[27] Marina Munkhoeva, Yermek Kapushev, Evgeny Burnaev, and Ivan Oseledets, "Quadrature-based features for kernel approximation," in *Advances in Neural Information Processing Systems*, 2018, pp. 9147–9156.

[28] Alaa Saade, Francesco Caltagirone, Igor Carron, Laurent Daudet, Angélique Drémeau, Sylvain Gigan, and Florent Krzakala, "Random projections through multiple optical scattering: Approximating kernels at the speed of light," in *Proceedings og IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2016, pp. 6215–6219.

[29] Ruben Ohana, Jonas Wacker, Jonathan Dong, Sébastien Marmin, Florent Krzakala, Maurizio Filippone, and Laurent Daudet, "Kernel computations from large-scale random features obtained by optical processing units," *arXiv preprint arXiv:1910.09880*, 2019.

[30] Danica J. Sutherland and Jeff Schneider, "On the error of random Fourier features," in *Conference on Uncertainty in Artificial Intelligence*, 2015, pp. 862–871.

[31] Zhu Li, Jean-Francois Ton, Dino Oglic, and Dino Sejdinovic, "Towards a unified analysis of random Fourier features," in *the 36th International Conference on Machine Learning*, 2019, pp. 3905–3914.

[32] Ali Rahimi and Benjamin Recht, "Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning," in *Advances in neural information processing systems*, 2009, pp. 1313–1320.

[33] Fuxin Li, Catalin Ionescu, and Cristian Sminchisescu, "Random Fourier approximations for skewed multiplicative histogram kernels," in *Joint Pattern Recognition Symposium*. Springer, 2010, pp. 262–271.

[34] Purushottam Kar and Harish Karnick, "Random feature maps for dot product kernels," in *International Conference on Artificial Intelligence and Statistics*, 2012, pp. 583–591.

[35] Andrea Vedaldi and Andrew Zisserman, "Efficient additive kernels via explicit feature maps," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 3, pp. 480–492, 2012.

[36] Quoc Le, Tamás Sarlós, and Alex J. Smola, "FastFood—approximating kernel expansions in loglinear time," in *International Conference on Machine Learning*, 2013, pp. 244–252.

[37] Jiyan Yang, Vikas Sindhwani, Haim Avron, and Michael Mahoney, "Quasi-Monte Carlo feature maps for shift-invariant kernels," in *International Conference on Machine Learning*, 2014, pp. 485–493.

[38] Bo Dai, Bo Xie, Niao He, Yingyu Liang, Anant Raj, Maria-Florina F Balcan, and Le Song, "Scalable kernel methods via doubly stochastic gradients," in *Advances in Neural Information Processing Systems*, 2014, pp. 3041–3049.

[39] Jeffrey Pennington, Felix Xinnan X. Yu, and Sanjiv Kumar, "Spherical random features for polynomial kernels," in *Advances in Neural Information Processing Systems*, 2015, pp. 1846–1854.

[40] Chang Feng, Qinghua Hu, and Shizhong Liao, "Random feature mapping with signed circulant matrix projection," in *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[41] Krzysztof Choromanski and Vikas Sindhwani, "Recycling randomness with structure for sublinear time kernel expansions," in *International Conference on Machine Learning*, 2016, pp. 2502–2510.

[42] Weiwei Shen, Zhihui Yang, and Jun Wang, "Random features for shift-invariant kernels with moment matching," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017, pp. 2520–2526.

[43] Yueming Lyu, "Spherical structured feature maps for kernel approximation," in *34th International Conference on Machine Learning*. JMLR.org, 2017, pp. 2256–2264.

[44] Shahin Shahrampour, Ahmad Beirami, and Vahid Tarokh, "On data-dependent random features for improved generalization in supervised learning," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018, pp. 4026–4033.

[45] Jian Zhang, Avner May, Tri Dao, and Christopher Re, "Low-precision random Fourier features for memory-constrained kernel approximation," in *22nd International Conference on Artificial Intelligence and Statistics*, 2019, pp. 1264–1274.

[46] Raj Agrawal, Trevor Campbell, Jonathan Huggins, and Tamara Broderick, "Data-dependent compression of random features for large-scale kernel approximation," in *22nd International Conference on Artificial Intelligence and Statistics*, 2019, pp. 1822–1831.

[47] Fanghui Liu, Xiaolin Huang, Yudong Chen, Jie Yang, and Johan A.K. Suykens, "Random Fourier features via fast surrogate leverage weighted sampling," in *Thirty-Fourth AAAI Conference on Artificial Intelligence*, 2020, pp. 4844–4851.

[48] Tamás Erdélyi, Cameron Musco, and Christopher Musco, "Fourier sparse leverage scores and approximate kernel learning," in *Advances in Neural Information Processing Systems*, 2020.

[49] Bharath K. Sriperumbudur and Zoltán Szabó, "Optimal rates for random Fourier features," in *Advances in Neural Information Processing Systems*, 2015, pp. 1144–1152.

[50] Jean Honorio and Yu-Jun Li, "The error probability of random Fourier features is dimensionality independent," *arXiv preprint arXiv:1710.09953*, 2017.

[51] Haim Avron, Michael Kapralov, Cameron Musco, Christopher Musco, Ameya Velingker, and Amir Zandieh, "Random Fourier features for kernel ridge regression: Approximation bounds and statistical guarantees," in *International Conference on Machine Learning*, 2017, pp. 253–262.

[52] Francis Bach, "On the equivalence between kernel quadrature rules and random feature expansions," *Journal of Machine Learning Research*, vol. 18, no. 1, pp. 714–751, 2017.

[53] Alessandro Rudi and Lorenzo Rosasco, "Generalization properties of learning with random features," in *Advances in Neural Information Processing Systems*, 2017, pp. 3215–3225.

[54] Eran Malach, Gilad Yehudai, Shai Shalev-Shwartz, and Ohad Shamir, "Proving the lottery ticket hypothesis: Pruning is all you need," *arXiv preprint arXiv:2002.00585*, 2020.

[55] Salomon Bochner, *Harmonic Analysis and the Theory of Probability*, Courier Corporation, 2005.

[56] I. J. Schoenberg, "Positive definite functions on spheres," *Duke Mathematical Journal*, vol. 9, no. 1, pp. 96–108, 1942.

[57] Alex J. Smola, Zoltan L. Ovari, and Robert C. Williamson, "Regularization with dot-product kernels," in *Advances in Neural Information Processing Systems*, 2001, pp. 308–314.

[58] Claus Müller, *Spherical harmonics*, vol. 17, Springer, 2006.

[59] Guang-Bin Huang, Qin-Yu Zhu, and Chee-Kheong Siew, "Extreme learning machine: theory and applications," *Neurocomputing*, vol. 70, no. 1-3, pp. 489–501, 2006.

[60] Youngmin Cho and Lawrence K Saul, "Kernel methods for deep learning," in *Advances in Neural Information Processing Systems*, 2009, pp. 342–350.

[61] Christopher K.I. Williams, "Computing with infinite networks," in *Advances in Neural Information Processing Systems*, 1997, pp. 295–301.

[62] Dan Hendrycks and Kevin Gimpel, "Gaussian error linear units (GELUs)," *arXiv preprint arXiv:1606.08415*, 2016.

[63] Amit Daniely, Roy Frostig, and Yoram Singer, "Toward deeper understanding of neural networks: The power of initialization and a dual view on expressivity," in *Advances In Neural Information Processing Systems*, 2016, pp. 2253–2261.

[64] Jaehoon Lee, Yasaman Bahri, Roman Novak, Samuel S Schoenholz, Jeffrey Pennington, and Jascha Sohl-Dickstein, "Deep neural networks as Gaussian Processes," in *International Conference on Learning Representations*, 2018.

[65] Lenaic Chizat, Edouard Oyallon, and Francis Bach, "On lazy training in differentiable programming," in *Advances in Neural Information Processing Systems*, 2019, pp. 2933–2943.

[66] Alberto Bietti and Julien Mairal, "On the inductive bias of neural tangent kernels," in *Advances in Neural Information Processing Systems*, 2019, pp. 12873–12884.

[67] Behrooz Ghorbani, Song Mei, Theodor Misiakiewicz, and Andrea Montanari, "Linearized two-layers neural networks in high dimension," *Annals of Statistics*, 2019.

[68] Alberto Bietti and Francis Bach, "Deep equals shallow for ReLU networks in kernel regimes," in *International Conference on Learning Representations*, 2021.

[69] Marc G. Genton, "Classes of kernels for machine learning: a statistics perspective," *Journal of Machine Learning Research*, vol. 2, pp. 299–312, 2001.

[70] Sami Remes, Markus Heinonen, and Samuel Kaski, "Non-stationary spectral kernels," in *Advances in Neural Information Processing Systems*, 2017, pp. 4642–4651.

[71] Jean-Francois Ton, Seth Flaxman, Dino Sejdinovic, and Samir Bhatt, "Spatial mapping with Gaussian processes and nonstationary Fourier features," *Spatial Statistics*, vol. 28, pp. 59–78, 2018.

[72] Akiva M Yaglom, *Correlation Theory of Stationary and Related Random Functions*, Springer-Verlag, 1987.

[73] Ninh Pham and Rasmus Pagh, "Fast and scalable polynomial kernels via explicit feature maps," in *ACM International Conference on Knowledge Discovery and Data Mining*, 2013, pp. 239–247.

[74] Michela Meister, Tamas Sarlos, and David Woodruff, "Tight dimensionality reduction for sketching low degree polynomial kernels," in *Advances in Neural Information Processing Systems*, 2019, pp. 9475–9486.

[75] Haim Avron, Huy Nguyen, and David Woodruff, "Subspace embeddings for the polynomial kernel," in *Advances in neural information processing systems*, 2014, pp. 2258–2266.

[76] David P Woodruff and Amir Zandieh, "Near input sparsity time kernel embeddings via adaptive sampling," in *International Conference on Machine Learning*, 2020, pp. 10324–10333.

[77] Fanghui Liu, Lei Shi, Xiaolin Huang, Jie Yang, and Johan A.K. Suykens, "A double-variational Bayesian framework in random Fourier features for indefinite kernels," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 8, pp. 2965–2979, 2020.

[78] Fanghui Liu, Xiaolin Huang, Yingyi Chen, and Johan A.K. Suykens, "Fast learning in reproducing kernel Kreĭn spaces via signed measures," in *International Conference on Artificial Intelligence and Statistics*, 2021, pp. 1–11.

[79] Ping Li, "Linearized GMM kernels and normalized random Fourier features," in *23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017, pp. 315–324.

[80] Krzysztof M. Choromanski, Mark Rowland, and Adrian Weller, "The unreasonable effectiveness of structured random orthogonal embeddings," in *Advances in Neural Information Processing Systems*, 2017, pp. 219–228.

[81] Alessandro Rudi, Daniele Calandriello, Luigi Carratino, and Lorenzo Rosasco, "On fast leverage score sampling and optimal learning," in *Advances in Neural Information Processing Systems*, 2018, pp. 5672–5682.

[82] Wei-Cheng Chang, Chun-Liang Li, Yiming Yang, and Barnabás Póczos, "Data-driven random Fourier features using Stein effect," in *26th International Joint Conference on Artificial Intelligence*, 2017, pp. 1497–1503.

[83] Aman Sinha and John C. Duchi, "Learning kernels with random features," in *Proceedins of Advances in Neural Information Processing Systems*, 2016, pp. 1298–1306.

[84] Chun-Liang Li, Wei-Cheng Chang, Youssef Mroueh, Yiming Yang, and Barnabas Poczos, "Implicit kernel learning," in *International Conference on Artificial Intelligence and Statistics*, 2019, pp. 2007–2016.

[85] Felix X. Yu, Sanjiv Kumar, Henry Rowley, and Shih Fu Chang, "Compact nonlinear maps and circulant extensions," *arXiv preprint arXiv:1503.03893*, 2015.

[86] Brian Bullins, Cyril Zhang, and Yi Zhang, "Not-so-random features," in *International Conference on Learning Representations*, 2018.

[87] Andrew Gordon Wilson and Ryan Prescott Adams, "Gaussian process kernels for pattern discovery and extrapolation," in *International Conference on Machine Learning*, 2013, pp. 1067–1075.

[88] Zichao Yang, Andrew Wilson, Alex J. Smola, and Le Song, "À la carte–learning fast kernels," in *Artificial Intelligence and Statistics*, 2015, pp. 1098–1106.

[89] Zheyang Shen, Markus Heinonen, and Samuel Kaski, "Harmonizable mixture kernels with variational Fourier features," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2019.

[90] Junier B. Oliva, Avinava Dubey, Andrew G. Wilson, Barnabás Póczos, Jeff Schneider, and Eric P. Xing, "Bayesian nonparametric kernel learning," in *International Conference on Artificial Intelligence and Statistics*, 2016, pp. 1078–1086.

[91] Mariusz Bojarski, Anna Choromanska, Krzysztof Choromanski, Francois Fagan, Cedric Gouy-Pailler, Anne Morvan, Nouri Sakr, Tamas Sarlos, and Jamal Atif, "Structured adaptive and random spinners for fast machine learning computations," in *Artificial Intelligence and Statistics*, 2017, pp. 1020–1029.

[92] Harald Niederreiter, *Random number generation and quasi-Monte Carlo methods*, vol. 63, SIAM, 1992.

[93] Tianbao Yang, Yu Feng Li, Mehrdad Mahdavi, Rong Jin, and Zhi Hua Zhou, "Nyström method vs random Fourier features: a theoretical and empirical comparison," in *Advances in Neural Information Processing Systems*, 2012, pp. 476–484.

[94] Bo Xie, Yingyu Liang, and Le Song, "Scale up nonlinear component analysis with doubly stochastic gradients," in *Advances in Neural Information Processing Systems*, 2015, pp. 2341–2349.

[95] Xiang Li, Bin Gu, Shuang Ao, Huaimin Wang, and Charles X. Ling, "Triply stochastic gradients on multiple kernel learning.," in *Conference on Uncertainty in Artificial Intelligence*, 2017, pp. 1–9.

[96] Krzysztof Choromanski, Mark Rowland, Wenyu Chen, and Adrian Weller, "Unifying orthogonal Monte Carlo methods," in *International Conference on Machine Learning*, 2019, pp. 1203–1212.

[97] Krzysztof Choromanski, Mark Rowland, Tamás Sarlós, Vikas Sindhwani, Richard Turner, and Adrian Weller, "The geometry of random features," in *International Conference on Artificial Intelligence and Statistics*, 2018, pp. 1–9.

[98] Xiaoyun Li and Ping Li, "Quantization algorithms for random fourier features," in *International Conference on Machine Learning*, 2021, pp. 6369–6380.

[99] Johann S. Brauchart and Peter J. Grabner, "Distributing many points on spheres: minimal energy and designs," *Journal of Complexity*, vol. 31, no. 3, pp. 293–326, 2015.

[100] Yueming LYU, Yuan Yuan, and Ivor Tsang, "Subgroup-based rank-1 lattice quasi-monte carlo," in *Advances in Neural Information Processing Systems*, 2020.

[101] Gwynne Evans, *Practical numerical integration*, Wiley New York, 1993.

[102] Alan Genz and John Monahan, "Stochastic integration rules for infinite regions," *SIAM Journal on Scientific Computing*, vol. 19, no. 2, pp. 426–439, 1998.

[103] Alan Genz and John Monahan, "A stochastic algorithm for high-dimensional integrals over unbounded regions with gaussian weight," *Journal of Computational and Applied Mathematics*, vol. 112, no. 1-2, pp. 71–81, 1999.

[104] Florian Heiss and Viktor Winschel, "Likelihood approximation by numerical integration on sparse grids," *Journal of Econometrics*, vol. 144, no. 1, pp. 62–80, 2008.

[105] Ayoub Belhadji, Rémi Bardenet, and Pierre Chainais, "Kernel quadrature with dpps," in *Advances in Neural Information Processing Systems*, 2019, pp. 1–11.

[106] Fanghui Liu, Xiaolin Huang, Yudong Chen, and Johan A.K. Suykens, "Towards a unified quadrature framework for large-scale kernel machines," *arXiv preprint arXiv:2011.01668*, 2020.

[107] François-Xavier Briol, Chris J Oates, Jon Cockayne, Wilson Ye Chen, and Mark Girolami, "On the sampling problem for kernel quadrature," in *International Conference on Machine Learning*, 2017, pp. 586–595.

[108] Bertrand Gauthier and Johan A.K. Suykens, "Optimal quadrature-sparsification for integral operator approximation," *SIAM Journal on Scientific Computing*, vol. 40, no. 5, pp. A3636–A3674, 2018.

[109] Yinsong Wang and Shahin Shahrampour, "A general scoring rule for randomized kernel approximation with application to canonical correlation analysis," *arXiv preprint arXiv:1910.05384*, 2019.

[110] Tong Zhang, "Learning bounds for kernel regression using effective data dimensionality," *Neural Computation*, vol. 17, no. 9, pp. 2077–2098, 2005.

[111] Francis Bach, "Sharp analysis of low-rank kernel matrix approximations," in *Conference on Learning Theory*, 2013, pp. 185–209.

[112] Hayata Yamasaki, Sathyawageeswar Subramanian, Sho Sonoda, and Masato Koashi, "Fast quantum algorithm for learning with optimized random features," in *Advances in Neural Information Processing Systems*, 2020, pp. 1–10.

[113] Ahmed Alaoui and Michael W Mahoney, "Fast randomized kernel ridge regression with statistical guarantees," in *Advances in Neural Information Processing Systems*, 2015, pp. 775–783.

[114] Daniele Calandriello, Alessandro Lazaric, and Michal Valko, "Distributed adaptive sampling for kernel matrix approximation," in *Artificial Intelligence and Statistics*, 2017, pp. 1421–1429.

[115] Corinna Cortes, Mehryar Mohri, and Afshin Rostamizadeh, "Two-stage learning kernel algorithms," in *International Conference on Machine Learning*, 2010, pp. 239–246.

[116] Trevor Campbell and Tamara Broderick, "Bayesian coreset construction via greedy iterative geodesic ascent," in *International Conference on Machine Learning*, 2018, pp. 698–706.

[117] Trevor Campbell and Tamara Broderick, "Automated scalable Bayesian inference via Hilbert coresets," *Journal of Machine Learning Research*, vol. 20, no. 1, pp. 551–588, 2019.

[118] Raffay Hamid, Ying Xiao, Alex Gittens, and Dennis Decoste, "Compact random feature maps," in *International Conference on Machine Learning*, 2014, pp. 19–27.

[119] Ali Rahimi and Benjamin Recht, "Uniform approximation of functions with random bases," in *Annual Allerton Conference on Communication, Control, and Computing*. IEEE, 2008, pp. 555–561.

[120] Mina Ghashami, Daniel J. Perry, and Jeff Phillips, "Streaming kernel principal component analysis," in *Artificial intelligence and statistics*, 2016, pp. 1365–1374.

[121] Bharath Sriperumbudur and Nicholas Sterge, "Statistical consistency of kernel PCA with random features," *arXiv preprint arXiv:1706.06296*, 2017.

[122] Enayat Ullah, Poorya Mianjy, Teodor Vanislavov Marinov, and Raman Arora, "Streaming kernel PCA with $\tilde{O}(\sqrt{n})$ random features," in *Advances in Neural Information Processing Systems*, 2018, pp. 7311–7321.

[123] Felipe Cucker and Dingxuan Zhou, *Learning theory: an approximation theory viewpoint*, vol. 24, Cambridge University Press, 2007.

[124] Ingo Steinwart and Christmann Andreas, *Support Vector Machines*, Springer Science and Business Media, 2008.

[125] Gilles Blanchard and Nicole Krämer, "Optimal learning rates for kernel conjugate gradient regression," in *Advances in Neural Information Processing Systems*, 2010, pp. 226–234.

[126] Andrea Caponnetto and Ernesto De Vito, "Optimal rates for the regularized least-squares algorithm," *Foundations of Computational Mathematics*, vol. 7, no. 3, pp. 331–368, 2007.

[127] John Shawe-Taylor, Chris Williams, Nello Cristianini, and Jaz Kandola, "On the eigenspectrum of the gram matrix and its relationship to the operator eigenspectrum," in *International Conference on Algorithmic Learning Theory*. Springer, 2002, pp. 23–40.

[128] Steve Smale and Ding-Xuan Zhou, "Learning theory estimates via integral operators and their approximations," *Constructive Approximation*, vol. 26, no. 2, pp. 153–172, 2007.

[129] Zheng-Chu Guo and Lei Shi, "Optimal rates for coefficient-based regularized regression," *Applied and Computational Harmonic Analysis*, vol. 47, no. 3, pp. 662–701, 2019.

[130] Shao-Bo Lin, Xin Guo, and Ding-Xuan Zhou, "Distributed learning with regularized least squares," *Journal of Machine Learning Research*, vol. 18, no. 1, pp. 3202–3232, 2017.

[131] Vladimir Koltchinskii, *Oracle Inequalities in Empirical Risk Minimization and Sparse Recovery Problems*, vol. 2033, Springer Science & Business Media, 2011.

[132] Peter L Bartlett, Olivier Bousquet, and Shahar Mendelson, "Local rademacher complexities," *Annals of Statistics*, vol. 33, no. 4, pp. 1497–1537, 2005.

[133] Zhu Li, Jean-Francois Ton, Dino Oglic, and Dino Sejdinovic, "Towards a unified analysis of random fourier features," *Journal of Machine Learning Research*, vol. 22, no. 108, pp. 1–51, 2021.

[134] Luigi Carratino, Alessandro Rudi, and Lorenzo Rosasco, "Learning with SGD and random features," in *Advances in Neural Information Processing Systems*, 2018, pp. 10212–10223.

[135] Ingo Steinwart and Clint Scovel, "Fast rates for support vector machines using Gaussian kernels," *Annals of Statistics*, vol. 35, no. 2, pp. 575–607, 2007.

[136] Shusen Wang, "Simple and almost assumption-free out-of-sample bound for random feature mapping," *arXiv preprint arXiv:1909.11207*, 2019.

[137] Yann Lecun, Leon Bottou, Yoshua Bengio, and Patrick Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*.

[138] Alex Krizhevsky and Geoffrey Hinton, "Learning multiple layers of features from tiny images," *Technical report, University of Toronto*, 2009.

[139] Gaëlle Loosli, Stéphane Canu, and Léon Bottou, "Training invariant support vector machines using selective sampling," *Large scale kernel machines*, vol. 2, 2007.

[140] Sanjeev Arora, Simon S Du, Zhiyuan Li, Ruslan Salakhutdinov, Ruosong Wang, and Dingli Yu, "Harnessing the power of infinitely wide deep nets on small-data tasks," in *International Conference on Learning Representations*, 2020.

[141] Sergey Ioffe and Christian Szegedy, "Batch normalization: accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*, 2015, pp. 448–456.

[142] Jia Deng, Wei Dong, Richard Socher, Li Jia Li, Kai Li, and Fei Fei Li, "Imagenet: A large-scale hierarchical image database," in *Proceedins of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[143] Trevor Hastie, Andrea Montanari, Saharon Rosset, and Ryan J. Tibshirani, "Surprises in high-dimensional ridgeless least squares interpolation," *arXiv preprint arXiv:1903.08560*, 2019.

[144] Song Mei and Andrea Montanari, "The generalization error of random features regression: Precise asymptotics and double descent curve," *arXiv preprint arXiv:1908.05355*, 2019.

[145] Tengyuan Liang, Alexander Rakhlin, and Xiyu Zhai, "On the multiple descent of minimum-norm interpolants and restricted lower isometry of kernels," in *Annual Conference on Learning Theory*, 2019, pp. 1–32.

[146] Fanghui Liu, Zhenyu Liao, and Johan A.K. Suykens, "Kernel regression in high dimensions: Refined analysis beyond double descent," in *International Conference on Artificial Intelligence and Statistics*, 2021, pp. 1–11.

[147] Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever, "Deep double descent: Where bigger models and more data hurt," in *International Conference on Learning Representations*, 2019.

[148] Mikhail Belkin, Siyuan Ma, and Soumik Mandal, "To understand deep learning we need to understand kernel learning," in *International Conference on Machine Learning*, 2018, pp. 541–549.

[149] Felipe Cucker and Steve Smale, "On the mathematical foundations of learning," *Bulletin of the American mathematical society*, vol. 39, no. 1, pp. 1–49, 2002.

[150] Zeyuan Allen-Zhu, Yuanzhi Li, and Yingyu Liang, "Learning and generalization in overparameterized neural networks, going beyond two layers," in *Advances in neural information processing systems*, 2019, pp. 6158–6169.

[151] Terence Tao, *Topics in random matrix theory*, American Mathematical Society, 2012.

[152] Jeffrey Pennington and Pratik Worah, "Nonlinear random matrix theory for deep learning," in *Advances in Neural Information Processing Systems*, 2017, pp. 2634–2643.

[153] Zhenyu Liao and Romain Couillet, "On the spectrum of random features maps of high dimensional data," in *International Conference on Machine Learning*, 2018, pp. 3063–3071.

[154] Zhenyu Liao, Romain Couillet, and Michael Mahoney, "A random matrix analysis of random fourier features: beyond the gaussian kernel, a precise phase transition, and the corresponding double descent," in *Neural Information Processing Systems*, 2020.

[155] Jimmy Ba, Murat A. Erdogdu, Taiji Suzuki, Denny Wu, and Tianzong Zhang, "Generalization of two-layer neural networks: an asymptotic viewpoint," in *International Conference on Learning Representations*, 2020, pp. 1–8.

[156] Stéphane d'Ascoli, Maria Refinetti, Giulio Biroli, and Florent Krzakala, "Double trouble in double descent: Bias and variance(s) in the lazy regime," *arXiv preprint arXiv:2003.01054*, 2020.

[157] Federica Gerace, Bruno Loureiro, Florent Krzakala, Marc Mézard, and Lenka Zdeborová, "Generalisation error in learning with random features and the hidden manifold model," in *International Conference on Machine Learning*, 2020, pp. 3452–3462.

[158] Ben Adlam and Jeffrey Pennington, "Understanding double descent requires a fine-grained bias-variance decomposition," in *Advances in neural information processing systems*, 2020.

[159] Oussama Dhifallah and Yue M Lu, "A precise performance analysis of learning with random features," *arXiv preprint arXiv:2008.11904*, 2020.

[160] Hong Hu and Yue M Lu, "Universality laws for high-dimensional learning with random features," *arXiv preprint arXiv:2009.07669*, 2020.

[161] Arthur Jacot, Berfin Şimşek, Francesco Spadaro, Clément Hongler, and Franck Gabriel, "Implicit regularization of random feature models," in *International Conference on Machine Learning*, 2020, pp. 4631–4640.

[162] Mikhail Belkin, Daniel Hsu, and Ji Xu, "Two models of double descent for weak features," *SIAM Journal on Mathematics of Data Science*, vol. 2, no. 4, pp. 1167–1180, 2020.

[163] Jason W Rocks and Pankaj Mehta, "Memorizing without overfitting: Bias, variance, and interpolation in over-parameterized models," *arXiv preprint arXiv:2010.13933*, 2020.

[164] Licong Lin and Edgar Dobriban, "What causes the test error? going beyond bias-variance via anova," *arXiv preprint arXiv:2010.05170*, 2020.

[165] Marc Mézard, Giorgio Parisi, and Miguel Angel Virasoro, *Spin glass theory and beyond: An Introduction to the Replica Method and Its Applications*, vol. 9, World Scientific Publishing Company, 1987.

[166] Christos Thrampoulidis, Samet Oymak, and Babak Hassibi, "Regularized linear regression: A precise analysis of the estimation error," in *Conference on Learning Theory*, 2015, pp. 1683–1709.

[167] Andrea Montanari, Feng Ruan, Youngtak Sohn, and Jun Yan, "The generalization error of max-margin linear classifiers: High-dimensional asymptotics in the overparametrized regime," *arXiv preprint arXiv:1911.01544*, 2019.

[168] Tengyuan Liang and Pragya Sur, "A precise high-dimensional asymptotic theory for boosting and min-ℓ_1-norm interpolated classifiers," *arXiv preprint arXiv:2002.01586*, 2020.

[169] Adel Javanmard, Mahdi Soltanolkotabi, and Hamed Hassani, "Precise tradeoffs in adversarial training for linear regression," in *Conference on Learning Theory*, 2020, pp. 2034–2078.

[170] Cosme Louart, Zhenyu Liao, and Romain Couillet, "A random matrix approach to neural networks," *The Annals of Applied Probability*, vol. 28, no. 2, pp. 1190–1248, 2018.
[171] Song Mei, Theodor Misiakiewicz, and Andrea Montanari, "Generalization error of random features and kernel methods: hypercontractivity and kernel matrix concentration," *arXiv preprint arXiv:2101.10588*, 2021.
[172] Gilad Yehudai and Ohad Shamir, "On the power and limitations of random features for understanding neural networks," in *Advances in Neural Information Processing Systems*, 2019, pp. 6594–6604.
[173] Yitong Sun, Anna Gilbert, and Ambuj Tewari, "On the approximation properties of random ReLU features," *arXiv preprint arXiv:1810.04374*, 2018.
[174] Jonathan Frankle and Michael Carbin, "The lottery ticket hypothesis: Finding sparse, trainable neural networks," in *International Conference on Learning Representations*, 2019.
[175] Hengyuan Hu, Rui Peng, Yu-Wing Tai, and Chi-Keung Tang, "Network trimming: A data-driven neuron pruning approach towards efficient deep architectures," *arXiv preprint arXiv:1607.03250*, 2016.
[176] Giacomo Meanti, Luigi Carratino, Lorenzo Rosasco, and Alessandro Rudi, "Kernel methods through the roof: Handling billions of points efficiently," in *Advances in Neural Information Processing Systems*, 2020.
[177] Lin Chen, Yifei Min, Mikhail Belkin, and Amin Karbasi, "Multiple descent: Design your own generalization curve," *arXiv preprint arXiv:2008.01036*, 2020.
[178] Denny Wu and Ji Xu, "On the optimal weighted ℓ_2 regularization in overparameterized linear regression," in *Advances in Neural Information Processing Systems*, 2020, pp. 1–11.
[179] Dmitry Kobak, Jonathan Lomond, and Benoit Sanchez, "The optimal ridge penalty for real-world high-dimensional data can be zero or negative due to the implicit ridge regularization," *Journal of Machine Learning Research*, vol. 21, no. 169, pp. 1–16, 2020.
[180] Behrooz Ghorbani, Song Mei, Theodor Misiakiewicz, and Andrea Montanari, "Limitations of lazy training of two-layers neural network," in *Advances in Neural Information Processing Systems*, 2019, pp. 9108–9118.