Artificial Intelligence Review (2021) 54:843–876
https://doi.org/10.1007/s10462-020-09880-z

Major advancements in kernel function approximation

Deena P. Francis[1] · **Kumudha Raimond**[1]

Published online: 1 August 2020

Abstract
Kernel based methods have become popular in a wide variety of machine learning tasks. They rely on the computation of kernel functions, which implicitly transform the data in its input space to data in a very high dimensional space. Efficient application of these functions have been subject to study in the last 10 years. The main focus was on improving the scalability of kernel based methods. In this regard, kernel function approximation using explicit feature maps have emerged as a substitute for traditional kernel based methods. Over the years, various advancements from the theoretical perspective have been made to explicit kernel maps, especially to the method of random Fourier features (RFF), which is the main focus of our work. In this work, the major developments in the theory of kernel function approximation are reviewed in a systematic manner and the practical applications are discussed. Furthermore, we identify the shortcomings of the current research, and discuss possible avenues for future work.

Keywords Kernel · Approximation · Explicit feature maps · Kernel function · Theoretical guarantees · Random Fourier features · Classification

1 Introduction

Kernel functions are crucial part of kernel based classification, regression and clustering tasks. Learning algorithms have successfully used the powerful concept of kernels in various applications (Chang and Wu 2015; Lee et al. 2004; Wang 2012). The major developments in the area of kernel based learning have focused on improving the empirical performance of learning algorithms, providing theoretical guarantees, improving scalability and demonstrating their applicability in various kinds of modern data. The focus on scalability has resulted in the merging of ideas from different areas like numerical linear algebra, functional analysis and approximation algorithms. Approximation methods have been traditionally used to improve the time complexity, while compromising on quality of the

✉ Deena P. Francis
deena.francis@gmail.com

Kumudha Raimond
kraimond@karunya.edu

[1] Department of Computer Science and Engineering, Karunya Institute of Technology and Sciences, Coimbatore, India

end result. Ideas such as these have also been useful in developing scalable kernel methods. It is the aim of this work to track all the major developments in the area of kernel function approximation.

In Sect. 2, a basic introduction to kernel functions and kernel function approximation is provided. The theoretical developments in shift invariant kernel approximation, in particular RFF are descibed in detail in Sect. 3. In Sect. 4 the important works in dot product kernel approximation are described. In Sect. 5, summarization and comparison of all past reasearch in kernel function approximation is provided. Apart from the theoretical advancements, many practical uses of kernel function approximations have been developed, which are discussed in Sect. 6. In Sect. 6, we also provide some comparison experiments regarding kernel function approximation and learning accuracy. Finally the open problems and possible directions of future research in this area are discussed in Sect. 7.

1.1 Main contributions

The main contributions of this paper are as follows:

- Discussion and analysis of the major advances in the theory of kernel function approximation.
- Review of practical applications in which the recent advancements have been used.
- Experiments on kernel function approximation error and learning performance of shift invariant kernel approximations.
- Discussion on some of the open problems and possible directions for their solutions.

An overview of the kernel function approximation advancements discused in this work is given in Fig. 1. The theoretical results of past research that are discussed in this work mainly focus on improving the entry-wise errors, time to apply kernel map, learning guarantees and bounding the number of parameters of the approximations. We also delve into the practical applications such as deep learning, kernel principal component analysis (KPCA), kernel ridge regression (KRR), kernel clustering and online learning. The experiments provided in this work also provide some insight into the effect of kernel approximation accuracy and learning performance.

2 Preliminaries

Throughout the text, boldface variables in capital letters denote matrices, and boldface variables in small letters denote vectors. The input matrices are of the *object-feature* (Mahoney 2011) type, where information about objects are described by the features. An example of such a matrix is a list of patient records in which the features like height, weight, age, symptom1, symptom2, ... of each patient is stored along the columns, and each row holds a particular patient's record.

2.1 Notations

$\mathbf{X}$ denotes the input matrix of size $n \times d$, where n is the number of rows and d is the number of columns. $\mathbf{x}_{i,j}$ denotes the entry in the ith row and jth column. k is the kernel function that computes the inner product between vectors in the feature space. The notation $\langle \mathbf{x}, \mathbf{y} \rangle$

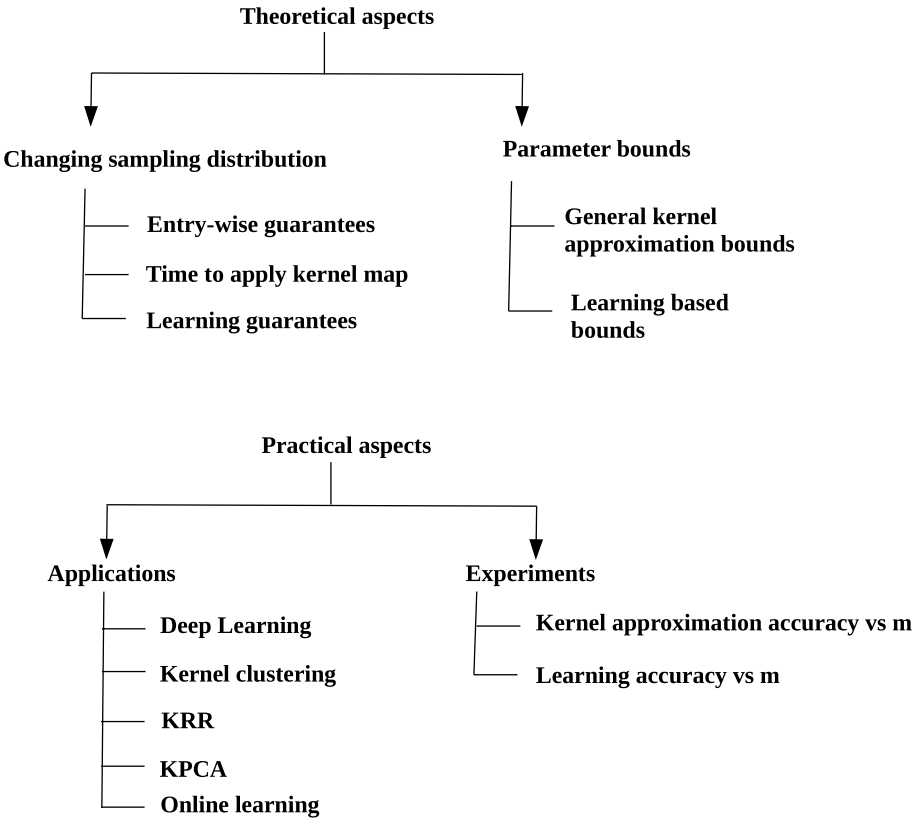


Fig. 1 Apsects of kernel function approximation discussed in this work

denotes the inner product between two vectors $\mathbf{x}$ and $\mathbf{y}$. $\mathbf{K}$ is the kernel matrix containing the inner products between all the n points of the input matrix, in the feature space, $\mathbf{K} \in \mathbb{R}^{n \times n}$. $|\mathbf{x}|$ is the Euclidean length of vector $\mathbf{x} \in \mathbb{R}^n$, where $|\mathbf{x}| = \sqrt{\sum_{i=1}^{n} |\mathbf{x}_i|^2}$. The Frobenius norm of $\mathbf{X}$ is defined as $\|\mathbf{X}\|_F = \sqrt{\sum_{i=1}^{n} \sum_{j=1}^{d} |\mathbf{x}_{i,j}|^2}$, and spectral norm is defined as $\sigma_{\max}(\mathbf{X})$, the maximum singular value of $\mathbf{X}$. Any matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ is positive semi-definite if its eigen values are non-negative or if $\forall x \in \mathbb{R}^n$, $\mathbf{x}^T \mathbf{A} \mathbf{x} \geq 0$. $L_p(\mathbf{X})$ denotes the space of functions that map $\mathbf{X}$ to either $\mathbb{R}$ or $\mathbb{C}$, where $\|\cdot\|_p < \infty$. For example, $L_2(\mathbf{X}) : \{f : \mathbf{X} \to \mathbb{R} \text{ or } \mathbb{C} : \|f\|_2 < \infty\}$.

2.2 Kernel functions

Let ϕ be a mapping from the input space to a very high dimensional space, F. Very high computational costs are incurred by applying ϕ directly on the data points. In addition suppose we wanted to compute $\langle \phi(\mathbf{x}), \phi(\mathbf{y}) \rangle$. A kernel function ($k$) can be used to perform the computation of dot products between the data points in F without explicitly computing $\phi(\mathbf{x})$ and $\phi(\mathbf{y})$, for any two data points $\mathbf{x}$ and $\mathbf{y}$ as shown in Eq. 1.

$$k(\mathbf{x}, \mathbf{y}) = \langle \phi(\mathbf{x}), \phi(\mathbf{y}) \rangle_F \tag{1}$$

An important theorem that relates kernel functions and a Reproducing Kernel Hilbert space (RKHS) is the famous Moore-Aronszajn theorem (Aronszajn 1950). It states that for any positive definite kernel, there exists a unique RKHS. Positive definite kernel functions could also be related to integral operators via Mercer's theorem. It states that any positive definite kernel can be written as a combination of the eigenfunctions and eigen values of integral operator, $T_k f(\cdot)$ defined using k as follows.

$$T_k f(\cdot) = \int_{\mathcal{X}} k(\cdot, \mathbf{x}) f(\mathbf{x}) d\mathbf{x}$$

Here $\mathcal{X}$ is any compact subset of $\mathbb{R}^n$, $f(\cdot) \in L_2(\mathcal{X})$.

$$k(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{\infty} \lambda_i \psi(\mathbf{x}) \psi(\mathbf{y})$$

where λ_i represents the eigen values of T_k and ψ_i represents the eigenfunctions of T_k.

There are basically two kernel functions of interest: shift invariant and dot product kernels. This categorization is based on the effect of translation on the kernel function. Shift invariant kernels are invariant to translation, whereas dot-product kernels are not invariant to translation.

2.2.1 Shift invariant kernels

Any kernel function k whose value depends only on the difference between the input vectors is called a shift invariant kernel.

$$k(\mathbf{x}, \mathbf{y}) = k(\mathbf{x} - \mathbf{y})$$

Some of the commonly used kernels that fall under this category are described below.
Gaussian kernel: It estimates the distance between two vectors by using a Gaussian function with width (standard deviation) σ.

$$k(\mathbf{x}, \mathbf{y}) = e^{\frac{-\|\mathbf{x}-\mathbf{y}\|^2}{2\sigma^2}} \tag{2}$$

Matern kernel: For a smoothness parameter α, the Matern kernel is defined as follows (Eq. 3).

$$k(\mathbf{x}, \mathbf{y}) = \frac{1}{2^{\alpha-1}\Gamma(\alpha)} \left(\frac{2\sqrt{\alpha}}{\theta} \|\mathbf{x} - \mathbf{y}\| \right)^{\alpha} \mathcal{J}_{\alpha} \left(\frac{2\sqrt{\alpha}}{\theta} \|\mathbf{x} - \mathbf{y}\| \right) \tag{3}$$

Here $\mathcal{J}_{\alpha}$ denotes a modified Bessel function of the first kind of order α and Γ denotes the Gamma function.

Laplacian kernel: It is a radial basis function kernel given as follows (Eq. 4). Here σ is the width (standard deviation) of the kernel.

$$k(\mathbf{x}, \mathbf{y}) = e^{\frac{-\|\mathbf{x}-\mathbf{y}\|}{\sigma}} \tag{4}$$

2.2.2 Dot product kernels

A kernel function whose result can be written as the dot product between the input vectors is called a dot product kernel. Some of the commonly used kernels that belong to this category are described below.

Linear kernel A linear kernel linearly scales one vector with another (Eq. 5).

$$k(\mathbf{x}, \mathbf{y}) = \mathbf{x}^T\mathbf{y} + c \tag{5}$$

A bias or shift, c is also sometimes added to the result.

Polynomial kernel It computes a degree q polynomial of the dot product of two vectors $\mathbf{x}$ and $\mathbf{y}$ as shown in Eq. 6.

$$k(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^T\mathbf{y} + c)^q \tag{6}$$

The constant c can be incorporated into the vectors and the equation simply becomes $k(\mathbf{x}, \mathbf{y}) = (\mathbf{x'}^T\mathbf{y'})^q$.

Sigmoid kernel This kernel function uses the tanh function (Eq. 7).

$$k(\mathbf{x}, \mathbf{y}) = \tanh(m\mathbf{x}^T\mathbf{y} + c) \tag{7}$$

where m and c are the slope and intercept terms respectively.

2.3 Kernel matrix

The kernel matrix, $\mathbf{K} \in \mathbb{R}^{n \times n}$ contains in its (i, j)th entry, the result of $k(\mathbf{x}_i, \mathbf{x}_j)$. Any kernel k satisfying Eq. 1, must have the property that its kernel matrix $\mathbf{K}$ is positive semi-definite (PSD). The kernel matrix can be used to determine the quality of the kernel function. Various metrics such as regularized risk, hyperkernels, negative log-posterior and feature space based kernel matrix evaluation measure were proposed in order to empirically measure the quality of the kernel.

2.4 Need for kernel approximation

Kernels are powerful techniques that have been used successfully in various applications of machine learning. Kernel functions are used to transform the given data to a high dimensional space, making it amenable to the effective application of learning algorithms such as classification or regression. Kernel functions are applied on all pairs of data points to obtain the kernel matrix $\mathbf{K}$. Kernel based methods are not scalable because of the time required to map n points using a kernel function scales as $O(n^2)$. In order to address this scalability issue, two main ideas have been explored in the literature of kernel based learning. The first idea deals with kernel matrix approximation, which is not discussed in this work. The previous works in this category used sampling (Williams and Seeger 2000; Drineas and Mahoney 2005; Cohen et al. 2015; Musco and Musco 2016) for approximating the kernel matrix. The second idea deals with the approximation of the kernel function, which is the topic of this work.

2.5 Kernel function approximation

The summary of some of the most important research done in kernel function approximation are shown in Fig. 2. The methods shown in this figure are the major algorithmic developments in the area of kernel function approximation.

First we define the problem of kernel function approximation.

Definition 1 (*Kernel function approximation*) Given a kernel function $f : \{\mathbf{x}, \mathbf{y}\} \to \mathbb{R}$ for any $\mathbf{x} \in \mathbf{X}, \mathbf{y} \in \mathbf{Y}, \mathbf{X}, \mathbf{Y} \in \mathbb{R}^{n \times d}$, find an $\tilde{f}$ such that the following is satisfied.

$$\tilde{f} = arg\, min_g \|f - g\|^2$$

The quantity $\|f - g\|^2 = \int_a^b (f(x) - g(x))^2$ is nothing but the inner product of function $f - g$ with itself. Here $[a, b]$ is any interval in which f is continuous.

The approximate function $\tilde{f}$ while being faster to compute must maintain guarantees with respect to quality. Bochner's theorem is a fundamental result that has been instrumental in bringing about novel ideas in the area of kernel function approximation.

3 Approximating shift-invariant kernels and RFF

In this section a key theorem in approximating shift-invariant kernels and the current research on RFF are discussed. In Sect. 3.3 we discuss the implications of changing the sampling distribution for approximating shift-invariant kernels and in Sect. 3.5 we discuss the implications of adding data-dependency.

Bochner's theorem is an important theorem that forms the building block for shift-invariant kernel approximation. A function $f : \mathbf{X} \times \mathbf{X} \to \mathbb{R}$ is positive definite if for any real

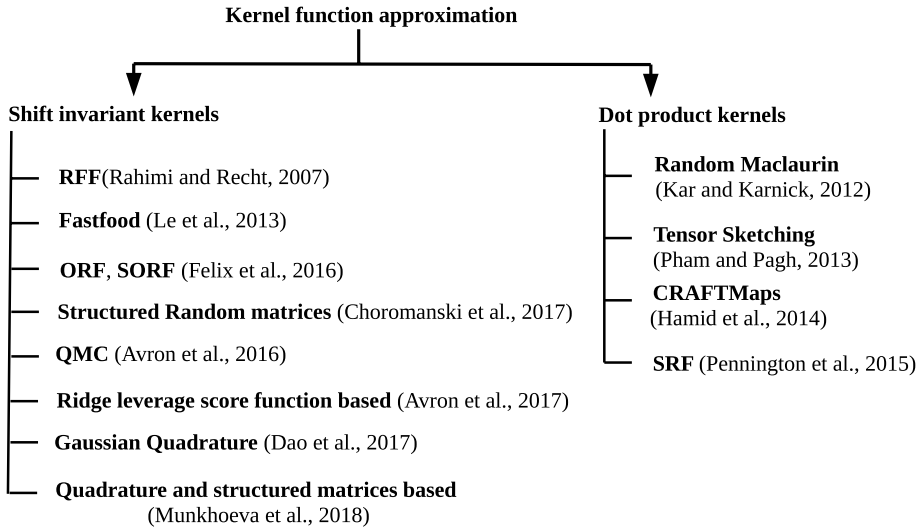


Fig. 2 Research in kernel function approximation

numbers $x_1, x_2, \ldots, x_n$, the matrix obtained by applying f on the difference between all (n^2) pairs of points is positive semi-definite.

Theorem 1 (Bochner's theorem) *Any continuous real valued function f is positive definite if and only if it is the Fourier transform of a positive measure.*

Consider the Gaussian kernel, $K(\mathbf{x}-\mathbf{y})$, and let $p(\boldsymbol{\omega}) = \frac{1}{\left(\sqrt{2\pi}\right)^m} e^{\frac{-\|\boldsymbol{\omega}\|_2^2}{2}}$ be its Fourier transform. Then by Bochner's theorem, $p(\boldsymbol{\omega})$ is a positive measure, and a probability distribution. Denoting F^{-1} with the inverse Fourier transform operator, we have:

$$\begin{aligned} K(\mathbf{x},\mathbf{y}) &= e^{\frac{-\|\mathbf{x}-\mathbf{y}\|}{2\sigma^2}} \\ &= F^{-1}(p(\boldsymbol{\omega})) \\ &= \int_{\mathbb{R}^d} p(\boldsymbol{\omega}) e^{jw^T(\mathbf{x}-\mathbf{y})} d\boldsymbol{\omega} \\ &= \mathbb{E}_{\boldsymbol{\omega}}\left[z(\mathbf{x}) z(\mathbf{y})^*\right] \end{aligned}$$

The last equation implies that $z(\mathbf{x})$ is an unbiased estimator of $K(\mathbf{x},\mathbf{y})$, and for $z(\mathbf{x}) = e^{j\omega^T\mathbf{x}}$ this is true.

Rahimi et al. (2007) proposed an explicit feature map, called RFF. It is the idea of explicitly mapping the data points to a low-dimensional space with the objective of approximating a shift-invariant kernel, k. This map is a direct consequence of applying Bochner's theorem. For any two points $\mathbf{x}$ and $\mathbf{y}$, $\mathbf{x},\mathbf{y} \in \mathbb{R}^d$, an explicit mapping function z is applied, such that the following is true.

$$K(\mathbf{x},\mathbf{y}) = \mathbb{E}_{\boldsymbol{\omega}}[z(\mathbf{x})^T z(\mathbf{y})] \tag{8}$$

The following formulations of z can be used.

$$z(\mathbf{x}) = [\cos(\boldsymbol{\omega}^T\mathbf{x} + \mathbf{b})] \tag{9}$$

$$z(\mathbf{x}) = [\cos(\boldsymbol{\omega}^T\mathbf{x})\, \sin(\boldsymbol{\omega}^T\mathbf{x})] \tag{10}$$

where $\boldsymbol{\omega} \in \mathbb{R}^d, \mathbf{b} \in \mathbb{R}^d, \boldsymbol{\omega} \sim p(\boldsymbol{\omega}), \mathbf{b} \sim \mathbb{N}(0, 2\pi)$. Interested readers are referred to the work of Sutherland and Schneider (2015) for details on comparison of the two formulations (Eqs. 9 and 10) in terms of error of kernel approximation. A different formulation Eq. 11 of RFF map, z was used by Avron et al. (2017). Here, $j = \sqrt{-1}$.

$$z(\mathbf{x}) = e^{-2\pi j\omega^T\mathbf{x}} \tag{11}$$

3.1 Bias and variance

The values returned by Eqs. 9, 10 and 11 are all scalars. In order to reduce variance of the approximation, one can resort to sampling m random vectors from the Fourier transform of the kernel function, $p(\boldsymbol{\omega})$ and stacking these m vectors to construct the features vector, $\mathbf{z} \in \mathbb{R}^d$. We refer to the parameter m as the *kernel mapping parameter*. The resulting maps are given below.

$$\mathbf{z}(\mathbf{x}) = \sqrt{\frac{2}{m}} \begin{bmatrix} \cos(\boldsymbol{\omega}_1^T \mathbf{x} + \mathbf{b}_1) \\ \cdots \\ \cos(\boldsymbol{\omega}_m^T \mathbf{x} + \mathbf{b}_m) \end{bmatrix} \tag{12}$$

$$\mathbf{z}(\mathbf{x}) = \sqrt{\frac{1}{m}} \begin{bmatrix} \cos(\boldsymbol{\omega}_1^T \mathbf{x}) \\ \sin(\boldsymbol{\omega}_1^T \mathbf{x}) \\ \cdots \\ \cos(\boldsymbol{\omega}_{m/2}^T \mathbf{x}) \\ \sin(\boldsymbol{\omega}_{m/2}^T \mathbf{x}) \end{bmatrix} \tag{13}$$

$$\mathbf{z}(\mathbf{x}) = \sqrt{\frac{1}{2m}} \begin{bmatrix} e^{-2\pi j \boldsymbol{\omega}_1^T \mathbf{x}} \\ \cdots \\ e^{-2\pi j \boldsymbol{\omega}_m^T \mathbf{x}} \end{bmatrix} \tag{14}$$

RFF formulations and kernels

The kernel defined by the first formulation of RFF (Eq. 12), denoted by K_{RFF1} is given as follows (Eq. 15).

$$K_{RFF1} = \frac{1}{m} \sum_{i=1}^{m} \cos(\boldsymbol{\omega}_i^T (\mathbf{x} - \mathbf{y})) + \cos(\boldsymbol{\omega}_i^T (\mathbf{x} + \mathbf{y} + 2b_i)) \tag{15}$$

The kernel defined by the second formulation of RFF (Eq. 13), denoted by K_{RFF2} is given as follows (Eq. 16).

$$K_{RFF2} = \frac{2}{m} \sum_{i=1}^{m/2} \cos(\boldsymbol{\omega}_i^T (\mathbf{x} - \mathbf{y})) \tag{16}$$

The kernel defined by the third formulation of RFF (Eq. 14), denoted by K_{RFF3} is given as follows (Eq. 17).

$$K_{RFF3}(\mathbf{x}, \mathbf{y}) = \frac{1}{2} \sum_{i=1}^{m} e^{-2\pi j \boldsymbol{\omega}_i^T \mathbf{x}} e^{2\pi j \boldsymbol{\omega}_i^T \mathbf{y}} = \frac{1}{2} \sum_{i=1}^{m} e^{-2\pi j \boldsymbol{\omega}^T (\mathbf{x} - \mathbf{y})} \tag{17}$$

Lemma 1 *The RFF kernels $K_{RFF1}, K_{RFF2}, K_{RFF3}$ are unbiased estimators of the Gaussian kernel.*

Proof By applying Bochner's theorem, we get, $\mathbb{E}[\cos(\boldsymbol{\omega}^T(\mathbf{x} - \mathbf{y}))] = \int_{\mathbb{R}^d} \cos(\boldsymbol{\omega}^T(\mathbf{x} - \mathbf{y})) d\boldsymbol{\omega} = k(\mathbf{x} - \mathbf{y})$. Hence, $\mathbb{E}[\cos(\boldsymbol{\omega}^T(\mathbf{x} - \mathbf{y}))] = k(\mathbf{x} - \mathbf{y})$. Finally, $\mathbb{E}[e^{-2\pi j \omega^T (\mathbf{x} - \mathbf{y})}] = k(\mathbf{x} - \mathbf{y})$. □

Lemma 2 *Let $\Delta = \mathbf{x} - \mathbf{y}$, the variance of the RFF kernels,*

(i) $\mathrm{Var}(K_{RFF1}(\mathbf{x}, \mathbf{y})) = \frac{1}{m}(1 + \frac{k(2(\Delta))}{2} - k(\Delta)^2$
(ii) $\mathrm{Var}(K_{RFF_2}(\mathbf{x}, \mathbf{y})) = \frac{1}{m}(1 + \frac{k(2\Delta)}{2} - k(\Delta)^2$
(iii) $\mathrm{Var}(K_{RFF3}(\mathbf{x}, \mathbf{y})) = \frac{1}{2m}(k(2(\Delta)) - (k(\Delta))^2$

Proof The proofs of (i) and (ii) can be found in Sutherland and Schneider (2015). Here we prove (iii). The covariance is computed first.

$$\begin{aligned}\text{Cov}(K_{RFF3}(\Delta), K_{RFF3}\Delta') &= \frac{1}{2m}\text{Cov}(e^{-2\pi j\boldsymbol{\omega}^T\Delta}, e^{-2\pi j\boldsymbol{\omega}^T\Delta'}) \\ &= \frac{1}{2m}\mathbb{E}[(e^{-2\pi j\boldsymbol{\omega}^T\Delta} - \mathbb{E}e^{-2\pi j\boldsymbol{\omega}^T\Delta})(e^{-2\pi j\boldsymbol{\omega}^T\Delta'}\mathbb{E}e^{-2\pi j\boldsymbol{\omega}^T\Delta'})] \\ &= \frac{1}{2m}[k(\Delta+\Delta') - k(\Delta)k(\Delta')^2]\end{aligned}$$

Thus the variance is given as follows.

$$\text{Cov}(K_{RFF3}(\Delta)) = \frac{1}{2m}(k(2\Delta) - k(\Delta)^2)$$

□

The third formulation (Eq. 14) has the lowest variance.

3.2 Relation to quadrature rules

Quadrature refers to the method used in numerical analysis to approximate definite integrals. The approximation is of the following form in which, a weighted sum of finite m samples of the integrand is computed.

$$\int_{\mathbb{R}^d} f(x)dx = \sum_{i=1}^{m} a_i f(x_i) \tag{18}$$

The optimal selection of weights a_i is crucial in obtaining a good approximation of the integral.

By Bochner's theorem, the shift invariant kernel function can be written as follows,

$$k(\mathbf{x},\mathbf{y}) = \int_{\mathbb{R}^d} p(\omega)e^{jw^T(\mathbf{x}-\mathbf{y})}d\omega \approx \frac{1}{m}\sum_{i=1}^{m} e^{j\omega_i^T(\mathbf{x}-\mathbf{y})} \tag{19}$$

The last equality (approximation) is due to the standard Monte-Carlo approximation to the integral. Furthermore, using the RFF method we can write the following.

$$k(\mathbf{x},\mathbf{y}) = \mathbb{E}[z(\mathbf{x})^T z(\mathbf{y})] \tag{20}$$

The problem of finding optimal weights for Eq. 18 is equivalent to finding the decomposition of k as given in Eq. 20 (Bach 2017).

3.3 Changing the sampling distribution

In RFF m vectors are sampled uniformly at random from the Fourier transform of the shift-invariant kernel. What happens when the sampling distribution is changed? The implications of this change are manifested in the following key areas.

1. Time taken to apply kernel map
2. Entry-wise guarantees
3. Learning guarantees

3.3.1 Reducing the time taken

The time required to compute the random features $\mathbf{z}(\mathbf{x})$, regardless of the type of formulation is $O(md)$, where d is the number of attributes (features) of $\mathbf{x}$. Many previous works have explored the idea of reducing the time taken to apply the kernel map.

The idea of using structured matrices for kernel function approximation was first proposed by Le et al. (2013) and their method is referred to as Fastfood in the literature. They proved that by choosing $\boldsymbol{\Omega}$ (the matrix of m vectors) in the following manner, the kernel mapping can be done in $O(m \log d)$ time.

$$\boldsymbol{\Omega} = \frac{1}{\sigma\sqrt{d}}\mathbf{D}_1\mathbf{H}\mathbf{D}_2\mathbf{P}\mathbf{H}\mathbf{D}_3 \tag{21}$$

where $\mathbf{D}_1$ is a random scaling diagonal matrix.
$\mathbf{H}$ is Walsh–Hadamard matrix.
$\mathbf{D}_2$ is a diagonal matrix with random Gaussian entries.
$\mathbf{P}$ is a permutation matrix with size $d \times d$ and entries 0 or 1.
$\mathbf{D}_3$ is diagonal matrix with entries $+1$ or -1.
σ is the width of the Gaussian function.
Felix et al. (2016) proposed Othogonal Random Features (ORF) where they replaced the Gaussian matrix used in RFF to draw random vectors $\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_m$ with an orthogonal matrix. This orthogonal matrix is constructed as the product of a uniformly distributed random orthogonal matrix $\mathbf{O}$ and a diagonal matrix $\mathbf{D}$, whose entries are drawn from the χ-distribution, having d degrees of freedom. They proved that with appropriate scaling, this construction is an unbiased estimator of the original Gaussian kernel. Furthermore, they provided empirical evidence of the superiority of their scheme when compared to RFF. However, ORF takes the exact same time ($O(md)$) to construct the random features for d-dimensional data points as RFF. They proposed further ideas which included structured matrices to reduce the time taken. By choosing the formualtion of Eq. 22 for $\boldsymbol{\Omega}$ better approximation gurantees and practical results were obtained. Here $\mathbf{H}$ is a normalized Walsh–Hadamard matrix and $\mathbf{D}_i$, for $i = 1, 2, 3$ are diagonal matrices, whose entries are drawn from the Rademacher distribution. This particular formulation is called Structured Orthogonal Random Features (SORF). The time for generating the features using SORF is $O(m \log d)$.

$$\boldsymbol{\Omega} = \frac{\sqrt{d}}{\sigma}\mathbf{H}\mathbf{D}_1\mathbf{H}\mathbf{D}_2\mathbf{H}\mathbf{D}_3 \tag{22}$$

The commonality of the two approaches is that both are trying to mimic the behavior of a Gaussian random matrix. In particular, carefully constructed structured matrices are used in place of the Gaussian matrix. Examples of such matrices are circulant matrices, Toeplitz and Hankel matrices. In order to see timing results of the comparison of these structured matrices, the readers are pointed to the results of Choromanski and Sindhwani (2016). The work of Choromanski et al. (2017) proposed Random Orthogonal Matrices (ROM) for approximating the Gaussian matrix. The idea is to use structured matrices for computational benefits and statistical advantages similar to orthogonal matrices. The Gaussian matrix of RFF can be replaced by $\boldsymbol{\Omega} = \sqrt{d}\Pi_{i=1}^{b}\mathbf{H}\mathbf{D}_i$. Here b is a parameter called block

size. A normalized Hadamard matrix $\mathbf{H}$ and a diagonal matrix $\mathbf{D}$ with ± 1 entries along its diagonal are used.

3.3.2 Entry-wise error guarantees

By entry-wise error guarantee, we mean that the error is given in terms of a particular entry (value) of the kernel matrix. The Hoeffding inequality can be used to bound the entry-wise error, regardless of the formulation of RFF used. Lemma 3 demonstrates such a bound, which is a loose bound on the entry-wise error of kernel approximation.

Lemma 3 $\Pr[|\mathbf{z}(\mathbf{x})^T\mathbf{z}(\mathbf{y}) - K(\mathbf{x},\mathbf{y})| \geq \epsilon] \leq 2e^{-m\epsilon^2/2}$

Proof The value of $\mathbf{z}(\mathbf{x})^T\mathbf{z}(\mathbf{y})$ lies between -1 and 1 due to the definition of $\mathbf{z}(\mathbf{x})$. By Hoeffding's inequality we have, $\Pr[|\mathbf{z}(\mathbf{x})^T\mathbf{z}(\mathbf{y}) - K(\mathbf{x},\mathbf{y})| \geq \epsilon] \leq 2e^{-\frac{2m\epsilon^2}{\sum_{i=1}^{m}(1+1)^2}}$. Simplifying this expression gives the result. □

A better (stronger) guarantee, given in Theorem 2 can be obtained by first proving it for a subset of $\mathbb{R}^d$ and then extending it to the whole space (Rahimi et al. 2007). The error is bounded over a region within ϵ radius in the $M \times M$ space, where M is any compact subspace of $\mathbb{R}^d$, $\sigma = \mathbb{E}[\omega^T\omega]$, $l = \text{diag}(M)$.

Theorem 2 (Rahimi et al. 2007) $\Pr[\sup_{\mathbf{x},\mathbf{y}\in M} |\mathbf{z}(\mathbf{x})^T\mathbf{z}(\mathbf{y}) - K(\mathbf{x},\mathbf{y})| \geq \epsilon] \leq 256\left(\frac{\sigma l}{\epsilon}\right)^2 \exp\left(-\frac{m\epsilon^2}{4(d+2)}\right)$
If $m = \Omega\left(d/\epsilon^2 \log \frac{\sigma l}{\epsilon}\right)$, *then with constant probability,* $\sup_{\mathbf{x},\mathbf{y}\in M} |\mathbf{z}(\mathbf{x})^T\mathbf{z}(\mathbf{y}) - K(\mathbf{x},\mathbf{y})| \leq \epsilon$

A tighter bound (Theorem 3) was later on derived by Sutherland and Schneider (2015). The constants $\alpha = \min(1, \sup_{\mathbf{x},\mathbf{y}\in\mathcal{X}} \frac{1}{2}(1 + k(2\mathbf{x}, 2\mathbf{y}) - 2k(\mathbf{x},\mathbf{y})^2 + \frac{2}{3}\epsilon)$ and $\beta = \left((\frac{d}{2})^{\frac{-d}{d+2}} + (\frac{d}{2})^{\frac{2}{d+2}}\right)2^{\frac{6d+2}{d+2}}$ were introduced.

Theorem 3 (Sutherland and Schneider 2015) *For the RFF formulation 2 in Eq.* 13, *the following bound exists.*

$$\Pr[\sup_{\mathbf{x},\mathbf{y}\in M} |\mathbf{z}(\mathbf{x})^T\mathbf{z}(\mathbf{y}) - K(\mathbf{x},\mathbf{y})| \geq \epsilon] \leq \beta\left(\frac{\sigma l}{\epsilon}\right)^{\frac{2}{1+\frac{2}{d}}} \exp\left(-\frac{m\epsilon^2}{8(d+2)\alpha}\right)$$

In the case of RFF formulation 1, Eq. 12, a looser bound exists as shown in Theorem 4. This is attributed to the noise introduced in it, and its corresponding kernel K_{RFF1} not being shift-invariant.

Theorem 4 (Sutherland and Schneider 2015) *For the RFF formulation 1 in Eq.* 12, $\alpha' = \min(1, \sup_{\mathbf{x},\mathbf{y}\in\mathcal{X}} \frac{1}{8}(2 + k(2\mathbf{x}, 2\mathbf{y}) - 2k(\mathbf{x},\mathbf{y})) + \frac{\epsilon}{6}$, $\beta' = \left(d^{\frac{-d}{1+d}} + d^{\frac{1}{1+d}}\right)2^{\frac{5d+1}{d+1}}3^{\frac{d}{d+1}}$ *the following bound exists.*

$$\Pr[\sup_{\mathbf{x},\mathbf{y}\in M} |\mathbf{z}(\mathbf{x})^T\mathbf{z}(\mathbf{y}) - K(\mathbf{x},\mathbf{y})| \geq \epsilon] \leq \beta'\left(\frac{\sigma l}{\epsilon}\right)^{\frac{2}{1+\frac{1}{d}}} \exp\left(-\frac{m\epsilon^2}{32(d+1)\alpha'}\right)$$

The finite-sample error bounds in Theorems 23 and 4 are far from optimal. Sriperumbudur and Szabó (2015) proposed the following error bound (Theorem 5), which holds for any m and assures sure convergence of the entry-wise error provided $Diam(M) = e^{o(m)}$.

Theorem 5 (Sriperumbudur and Szabó 2015) $\sup_{\mathbf{x},\mathbf{y}\in M} |\mathbf{z}(\mathbf{x})^T\mathbf{z}(\mathbf{y}) - K(\mathbf{x},\mathbf{y})| = O\left(|M|\sqrt{\frac{\log m}{m}}\right)$

This bound can be further improved by using McDiarmid's inequality, symmetrization lemma (Kulis and Grauman 2012, Proposition 7.10) and Dudley's entropy integral (Chaudhuri et al. 2011, Eq. 4.4). The following bound (Theorem 6) is obtained as a consequence.

Theorem 6 (Sriperumbudur and Szabó 2015) $\sup_{\mathbf{x},\mathbf{y}\in M} |\mathbf{z}(\mathbf{x})^T\mathbf{z}(\mathbf{y}) - K(\mathbf{x},\mathbf{y})| = O\left(\sqrt{\frac{\log |M|}{m}}\right)$

Thus using $m = O(\epsilon^{-2}\log|M|)$, we get the approximation accuracy of ϵ (as in Theorem 2).

Using the RFF method (formulation 3, Eq. 14) each entry of the kernel matrix, $\mathbf{K}$ can be approximated within $\pm\epsilon$. Expected maximum error, l_2 error bounds also exists for RFF (Sutherland and Schneider 2015).

The idea of using low-discrepancy sequences by Avron et al. (2016) was proven to have better asymptotic error than RFF. They showed that the error of their method converges at the rate of $O((\log m)^d/m)$ using specially designed $\boldsymbol{\omega}_i$, $i = 1, \ldots, m$. Avron et al. (2016) proposed using a 'specially designed sequence' in order to construct the vectors ω_i, $i = 1, \ldots, m$. This sequence is designed with the purpose of minimizing the following error for any function f any $\mathbf{x} \in [0,1]^d$.

$$v(f) = \int_{[0,1]^d} f(\mathbf{x})d\mathbf{x} - \sum_{\boldsymbol{\omega}\in\boldsymbol{\Omega}} f(\omega)$$

The sequences are generated deterministically and the have low-discrepancy. The discrepancy of a sequence is its degree of deviation from uniformity (Avron et al. 2016). Examples of such sequences include Halton, Niederriter, Faure and Sobol sequences. Using such sequences ensure that the error of their method converges at the rate of $O((\log m)^d/m)$. Their method is referred in the literature as Quasi-Monte Carlo (QMC) method. Even though the error of their method appears to be asymptotically better than that of standard RFF, it requires m to be quite large, in fact exponential in d to be useful.

Later, Dao et al. (2017) proposed the idea of deterministic sampling for kernel approximation using tools such as Gaussian quadrature. A key idea in their work is the data-adaptive method for picking a quadrature for a given number of samples, which was demonstrated to have much smaller approximation errors when compared to RFF. By assuming that the distribution of $\boldsymbol{\omega}_i$ is sub-Gaussian, one can achieve the following error bound. Their method is referred to as Gaussian Quadrature (GQ). Possible methods for the construction of quadrature rules that satisfy the following bound (Theorem 7) are given in the work of Dao et al. (2017), which include dense, sparse, and reweighted grid quadratures.

Theorem 7 (Dao et al. 2017) *If k is a kernel with sub-Gaussian spectrum, and $\tilde{k}$ is its estimate found using some quadrature rule, with weights whose sum is at most some even*

degree of L, b, a parameter of the sub-Gaussian. Then, for a region M with diameter l, *the following is true.* $|k(\mathbf{x},\mathbf{y}) - \tilde{k}(\mathbf{x},\mathbf{y})| \leq 3\left(\frac{ab^2l^2}{L}\right)^{L/2}$

Besides the improved theoretical bounds, their experimental results with approximations to the sparse ANOVA kernel for classification tasks showed improved accuracy over RFF based methods.

By defining special quadrature rules, called spherical-radial quadrature rules and using structured matrices better approximations to the kernel function can be obtained (Munkhoeva et al. 2018). In this case, the following error guarantee (Theorem 8) can be obtained.

Theorem 8 $\Pr[\sup_{\mathbf{x},\mathbf{y}\in M} |\mathbf{z}(\mathbf{x})^T\mathbf{z}(\mathbf{y}) - K(\mathbf{x},\mathbf{y})| \geq \epsilon] \leq \beta''\left(\frac{\sigma_p l\kappa\mu}{\epsilon}\right)^{\frac{2d}{d+1}} \exp\left(-\frac{m\epsilon^2}{8M^2(d+1)}\right)$ *here* $\beta'' = \left(d^{-\frac{d}{d+1}} + d^{\frac{1}{d+1}}\right)2^{\frac{6d+1}{d+1}}\left(\frac{d}{d+1}\right)^{\frac{d}{d+1}}$.
If $m = \Omega\left(\frac{M^2(d+1)}{\epsilon^2}\left[2(1+1/d)^{-1}\log\sigma_p l\kappa\mu\epsilon^{-1} + \log\beta''\delta^{-1}\right]\right)$, $\sup_{\mathbf{x},\mathbf{y}\in M} |\mathbf{z}(\mathbf{x})^T\mathbf{z}(\mathbf{y}) - K(\mathbf{x},\mathbf{y})| \leq \epsilon$

The connection between structured matrices based approach of ORF and spherical-radial quadrature rules was also proved in their work. They also demonstrated improved empirical performance of their approach on various classification and regression problems. They used butterfly matrices, and hence we refer to their work as Butterfly based quadrature (BQ).

The standard RFF approach (formulation of Eq. 14), when used to construct the kernel matrix is equivalent to sampling according to the column norms (Avron et al. 2017; Rudi and Rosasco 2017; Bach 2017). Other efficient score based sampling methods exist in the literature, one of which is the ridge leverage scores. Avron et al. (2017) focused on kernel approximation bounds of the following form (Eq. 23). Here $\lambda > 0$, $\mathbf{Z} \in \mathbb{R}^{n\times m}, \mathbf{Z} = [\mathbf{z}_1^*, \ldots, \mathbf{z}_m^*]$.

$$(1-\epsilon)(\mathbf{K} + \lambda\mathbb{I}) \leq (\mathbf{Z}\mathbf{Z}^* + \lambda\mathbb{I}) \leq (1+\epsilon)(\mathbf{K} + \lambda\mathbb{I}) \tag{23}$$

The above formulation of kernel approximation is significant in developing learning guarantees because it includes a regularization parameter (λ), which is prevalent in many learning algorithms. Besides, as demonstrated in their work, such bounds are useful in proving risk bounds for approximate KRR. For $\tilde{\mathbf{z}}(\boldsymbol{\omega}) = e^{-2\pi j\mathbf{x}_i^T\boldsymbol{\omega}}$, the ridge leverage score function defined as follows (Eq. 24).

$$\tau(\boldsymbol{\omega}) = p(\boldsymbol{\omega})\tilde{\mathbf{z}}(\boldsymbol{\omega})^*(\mathbf{K} + \lambda\mathbb{I})^{-1}\tilde{\mathbf{z}}(\boldsymbol{\omega}) \tag{24}$$

If ω were sampled according to the above equation, then it is possible to achieve bounds of Eq. 23 for $s = o(n)$. But, practical construction of the ridge leverage score function is not feasible. So instead, approximations to it can be considered. Avron et al. (2016) designed such an approximate function and subsequently proved better error bounds than RFF, but for m which must be exponential in d.

3.3.3 Learning

Kernel function approximation using RFF has implications in many algorithms like Kernel Ridge Regression (KRR), Kernel Principal Component Analysis (KPCA) etc. Using the

matrix Bernstein inequality (Mackey et al. 2014), one can bound the kernel matrix approximation error as follows (Theorem 9).

Theorem 9 (Lopez-Paz et al. 2014) *Let the original kernel matrix be* $\mathbf{K}$ *and the kernel approximation be* $\tilde{\mathbf{K}} = \frac{1}{m}\mathbf{z}(\mathbf{X})^T\mathbf{z}(\mathbf{X})$, *then the following is true.*

$$\mathbb{E}\|\tilde{\mathbf{K}} - \mathbf{K}\|_2 \leq \frac{2nlog(n)}{m} + \sqrt{\frac{3n^2 \log n}{m}}$$

While, Lopez-Paz et al. (2014) used the expectation bound of the matrix Bernstein inequality, Ghashami et al. (2016) used the probability bound to upper bound the kernel matrix approximation error and obtained the following bound (Theorem 10).

Theorem 10 (Ghashami et al. 2016) *For* $m = O(\epsilon^{-2}\log(n/\delta))$, *the following bound is admitted by RFF (formulation in Eq. 12) for probability at least* $1 - \delta$.

$$\|\mathbf{K} - \tilde{\mathbf{K}}\|_2 \leq \epsilon n \tag{25}$$

We prove similar bounds for RFF formulation in Eqs. 12 and 13 in Theorem 11. For the formulation of Eq. 14, we prove that such bounds do not exist, but using a slight modification of the definition of $\mathbf{Z}$, similar bounds can be proved as shown in Theorem 12. The following lemma (Lemma 4) is true because of the definition of $\mathbf{Z}$ using Eqs. 12 and 13. Similarly, Lemma 5 is also true because of the definition of $\mathbf{Z}$ using Eq. 14.

Lemma 4 *For RFF formulations of Eqs. 12 and 13, and* $\mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_m]$, *the following bound is true.*

$$\|\mathbf{Z}\|_F \leq 2n$$

Lemma 5 *For RFF formulation of Eq. 14, the following bound exists.* $\|\mathbf{Z}\|_F \leq mn$

Theorem 11 *The bound of Theorem 10 is satisfied for RFF formulation in Eq. 13, whereas for Eq. 14, it is not satisfied.*

Proof For formulation 2, using Lemma 4 in the proof of Lemma 3.1 in Ghashami et al. (2016), we get the desired result. In the case of formulation 3, Lemma 5 implies that the bound on kernel matrix approximation cannot be satisfied. □

Theorem 12 *By using the following modified formulation of Eq. 14, the bound of Theorem 10 is satisfied.*

$$\mathbf{z}(\mathbf{x}) = \sqrt{\frac{1}{2m}}\begin{bmatrix} e^{-2\pi j\boldsymbol{\omega}_1^T\mathbf{x}} \\ \cdots \\ e^{-2\pi j\boldsymbol{\omega}_m^T\mathbf{x}} \end{bmatrix} \tag{26}$$

Proof By this particular formulation we get $\|\mathbf{Z}\|_F \leq 2n$, and thus by following the proof of Lemma 3.1 in Ghashami et al. (2016), the result is obtained. □

It can easily be verified that, the new formulation given in Eq. 26 yields a kernel that is an unbiased estimator of the Gaussian kernel.

KPCA: The KPCA algorithm aims to find Principal Components (PCs) from a kernel transformed data. More precisely, given input data $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$, the aim is find a matrix $\mathbf{V}$ with orthonormal columns such that its columns capture the maximum variance of the data in the kernel space. So, initially a kernel function such as a Gaussian kernel (Eq. 2) is applied on all pairs of points in the given data. The kernel matrix is formed as a result of applying the kernel function. The eigen decomposition of this matrix gives $\mathbf{V}$. When the computation of PCs needs to be done in a streaming manner, a matrix sketching algorithm such as Frequent Directions (FD) (Liberty 2013) or its variants (Ghashami et al. 2014; Francis and Raimond 2018a) can be used.

Ghashami et al. (2016) showed that applying RFF (formulation 1, Eq. 12) and finding the matrix $\mathbf{V}$ using FD admits the following guarantee. Here $\tilde{\mathbf{K}} = \mathbf{Z}^T\mathbf{V}\mathbf{V}^T\mathbf{Z}$ and $\hat{\mathbf{K}} = \mathbf{Z}\mathbf{Z}^T$.

Lemma 6 (Ghashami et al. 2016) $\|\mathbf{K} - \hat{\mathbf{K}}_k\|_F^2 \leq \|\mathbf{K} - \tilde{\mathbf{K}}_k\|_F + \epsilon\sqrt{k}n$

KRR: In KRR, given training set, $(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n) \in \mathcal{X} \times \mathcal{Y}$, $\mathcal{X} \in \mathbb{R}^d, \mathcal{Y} \in \mathbb{R}$ and a regularization parameter $\lambda > 0$, the estimate of response variable, y is to be obtained, where

$$y = f(\mathbf{x}) = \sum_{i=1}^{n} c_i k(\mathbf{x}_i, \mathbf{x})$$

Here $\mathbf{c} = (c_1, \ldots, c_n)^T$ is the solution to the following.

$$\mathbf{y} = (\mathbf{K} + \lambda\mathbb{I})\mathbf{c}$$

The performance of the KRR estimator is analyzed using the risk operator, $\mathcal{R}$. If the empirical estimator is f and f^* is some other KRR estimator, then,

$$\mathcal{R}(f) = \mathbb{E}\left[\sum_{j=1}^{n}(f(\mathbf{x}_i) - f^*(\mathbf{x}_i))^2\right]$$

Let, $f : \mathcal{X} \to \mathbb{R}$, b_is are the normal random variables (noise terms) with standard deviation ν. If it is assumed that the response variable is of the following form.

$$y_i = f^*(\mathbf{x}_i) + b_i$$

If $\mathbf{f} \in \mathbb{R}^n$ is the vector whose jth entry is $f(\mathbf{x}_j)$, then the risk of the KRR estimate obtained using the modified RFF of Avron et al. (2017) can be bounded as follows (Theorem 13). Here, $\hat{\mathcal{R}}_{\mathbf{K}}(\mathbf{f})$ is an upper bound on the actual KRR estimator, $\mathcal{R}(\bar{f})$.

Theorem 13 (Avron et al. 2017) *Let $\tilde{f}$ be the KRR estimator obtained using $\tilde{k}$, then* $\mathcal{R}(\tilde{k}) \leq \frac{1}{(1-\Delta)}\hat{R}_{\mathbf{K}}(\mathbf{f}) + \frac{\Delta}{(1+\Delta)}\frac{\sigma^2\text{rank}(\tilde{\mathbf{K}})}{n}$

3.4 Bounding the number of samples

The number of samples (m) used has a direct effect on the accuracy of the kernel approximation. There are a few works that focus on bounding m with respect to kernel approximation bounds and/or learning guarantees.

If the goal is to achieve a regularized kernel approximation bound (Eq. 23) then, Avron et al. (2017) provided an upper bound of $o(n)$ by sampling from a ridge leverage score

distribution. A worse bound of $O(e^d)$ is obtained for m when trying to achieve similar kernel approximation bounds using an approximate ridge leverage score sampling. As described in Sect. 3.3.2, sampling from the true ridge leverage score distribution is infeasible, hence the use of its approximation. From a theoretical viewpoint, better kernel approximation guarantees (when compared to RFF) can be obtained by this method, but at the cost of requiring extremely large number of samples, which is undesirable. This bound for m is also undesirable from a practical viewpoint because datasets often have very large values of d.

The Gaussian quadrature method of Dao et al. (2017) also provides bounds on m. In order that Theorem 7 is satisfied for any $\gamma > 0$, and fixed d, the sample size, $m = O\left(\frac{1}{\epsilon^\gamma}\right)$. This bound is independent of d, but dependent on ϵ^{-1}. So, if ϵ is very small, then m would need to be very large.

3.4.1 Bounding the number of samples for learning tasks

Achieving good kernel matrix approximation accuracy alone is not enough to achieve good prediction accuracy (Bach 2013; Damodaran et al. 2017). Often, error guarantees with respect to learning tasks such as classification or regression are also desirable. In addition to developing such learning guarantees, bounds on m are also provided in conjuction. In order to measure the learning algorithm's performance, the risk operator can be used. Let $\mathbf{x}$ denote an instance and y denote its label. The actual risk of a function f is given as

$$\mathcal{R}[f] = \mathbb{E}[l(f(\mathbf{x}), y)]$$

where l is the loss function, and the expectation is taken over the probability distribution followed by $(\mathbf{x}, y)$. Examples of loss functions include square-absolute difference, hinge-loss etc. Generally, these loss functions are required to be continuous. By solving an empirical risk minimization problem, Rahimi and Recht (2009) showed that it is possible to bound the difference in risk $\mathcal{R}[f]$ and $\mathcal{R}[\hat{f}]$, where $\hat{f}$ is the function returned by the learning algorithm that uses RFF maps to predict the label y. The *excess risk* is bounded as follows.

$$\mathcal{R}[\hat{f}] - \mathcal{R}[f] \leq O\left(\frac{1}{\sqrt{m}}\right)$$

Cesa-Bianchi et al. (2004) proved convergence bounds for the risk of a general online learning algorithm. If we were to use RFF in such an online learning setting, then their bounds indirectly imply that $O(n)$ RFF samples are needed. If we require a risk bound of at most $O(1/m)$ as opposed to $O(1/\sqrt{m})$, then as per the facts given above, one would need $O(n^2)$ RFF samples. As a workaround, Lin et al. (2014) made use of regularized loss functions that are strongly convex and smooth. A regularized loss function takes the following form, where λ is the regularization parameter.

$$l_{reg} = \frac{\lambda}{2}\|f\|_{\mathcal{F}} + l(f(\mathbf{x}_i), y_i)$$

By optimizing the regularized loss function instead of the original loss function, the risk bound of the algorithm can be improved significantly. Concretely, in order to achieve an excess risk bound of $O(\log(n)/n)$, $O([c_l]^2)$ RFF samples suffice, where c_l is the conditional number of the regularized loss function. This constant $c_l = \beta/\lambda$, where,

$$|l''(f(\mathbf{x}, y), y)| \leq \beta$$

It is desirable to set the value of λ to be reasonably large, otherwise, the number of samples required increases. But, the choice of λ depends on the nature of the classification task. Thus this sample bound is not tight.

Besides the problem of a loose sample bound, the assumption that the training examples are independently and identically distributed (i.i.d.) proves to be an unrealistic scenario. In practice, training examples tend to be non i.i.d. By assuming that the training examples are being generated from an unknown mixing process, it is possible to obtain a risk bound of $O(\log(n)/n + \alpha)$, where α is a parameter of the mixing process (Hu et al. 2015). Consequently, the sample bound is $O(c_2)$, where c_2 is a constant which depends on the regularization parameter λ.

3.5 Adding data dependency

The samples in RFF are drawn from the distribution followed by the Fourier transform of the chosen kernel. Thus it is independent of the data. As a consequence, and as observed earlier, the number of samples required to achieve good prediction accuracy needs to be large. But, this is contrary to the notion of reducing the number of dimensions. It has been observed that adding a data dependency to RFF provides the benefits of providing smaller number of samples and good prediction accuracy (Sinha and Duchi 2016; Shahrampour et al. 2018; Damodaran et al. 2017). This idea is derived from the area of kernel learning, which describes a broad set of techniques to identify the best kernel for a particular learning problem. In fact, Sinha and Duchi (2016) combined ideas from kernel learning and RFF to learn a kernel for a supervised learning task. By solving an optimization problem, the best random features are selected. An improvement of this scheme involves the employment of a special scoring function to select the best random features for the attaining the best fit for the particular task (Shahrampour et al. 2018). Both the methods of Sinha and Duchi (2016) and Shahrampour et al. (2017) find the best kernel parameters by solving an optimization problem. The advantage of such a method is that lesser number of random features are needed to achieve the required goal, when compared to methods that directly use a fixed kernel. However, both these methods have computational overheads in the pre-processing stage. These overheads can be computationally expensive when d is large.

It is also possible to add dependency in an unsupervised manner by using stochastic gradient descent for learning the random features that minimize the error between original kernel and the estimated kernel (Damodaran et al. 2017). This idea is useful when the required downstream task is unsupervised. Practical experiments performed by Damodaran et al. (2017) indicated that their method achieves lower kernel approximation error and better accuracy than RFF, ORF and Nyström methods.

4 Approximation of dot product kernels

A polynomial kernel function is of the form,

$$K(\mathbf{x}, \mathbf{y}) = (a + \langle \mathbf{x}, \mathbf{y} \rangle)^p$$

where a is a bias term, and p is the degree of the polynomial.

It is possible to approximate the polynomial kernel by using a random Maclaurin expansion of the kernel function based on a theorem by Schoenberg (1938).

Theorem 14 (Schoenberg 1938) *A function $f : [-1, 1] \to \mathbb{R}$ is a positive definite kernel defined as $K(x, y) = f(\langle x, y \rangle)$ iff it is an analytic function with Maclaurin expansion using non-negative coefficients $a_i \geq 0$, $f(x) = \sum_{i=0}^{\infty} a_i x^i$.*

Kar and Karnick (2012) proved that a feature map, which we refer to as Random Maclaurin map is of the following form (Eq. 27).

$$z(\mathbf{x}) = \sqrt{a_N p^{N+1}} \prod_{i=1}^{N} \boldsymbol{\omega}_i^T \mathbf{x} \tag{27}$$

can be used to form the polynomial kernel. Here, $\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_N$ are Rademacher random variables, $N \in \mathbb{W}$ (set of whole numbers) and $\mathbb{P}(N = n) = \frac{1}{p^{n+1}}$. In order to improve the variance, m random vectors $\boldsymbol{\omega}$ can be applied on $\mathbf{x}$ and concatenated to form $\mathbf{Z}(x)$ (Eq. 28).

$$\mathbf{Z}(x) = \sqrt{\frac{1}{m}} \begin{bmatrix} \sqrt{a_N p^{N+1}} \prod_{i=1}^{N} \boldsymbol{\omega}_i^{(1)T} \mathbf{x} \\ \sqrt{a_N p^{N+1}} \prod_{i=1}^{N} \boldsymbol{\omega}_i^{(2)T} \mathbf{x} \\ \cdots \\ \sqrt{a_N p^{N+1}} \prod_{i=1}^{N} \boldsymbol{\omega}_i^{(m)T} \mathbf{x} \end{bmatrix} \tag{28}$$

Lemma 7 (Kar and Karnick 2012) *(Unbiasedness of dot product map) If the feature map $Z : \mathbb{R}^d \to \mathbb{R}$ constructed as in Eq. 28, then $\mathbb{E}[Z(\mathbf{x})Z(\mathbf{y})] = K(\mathbf{x}, \mathbf{y})$.*

Pham and Pagh (2013) proposed Tensor Sketching for approximating polynomial kernels by combining Count Sketch algorithm (Charikar et al. 2002) and Fast Fourier Transform (FFT). Their work utilized the equivalence between tensor product (outer product) as an explicit feature mapping and a polynomial kernel. The tensor product is then efficiently computed using convolution of Count Sketches. The idea is to construct Count Sketches (application of Count Sketch algorithm to the individual vectors) from the input vectors and then computing their outer products. This outer product of the Count Sketches can be computed efficiently using the FFT algorithm. The total cost of applying kernel mapping to a vector of dimension d using this method is $O(p(d + m \log(m)))$.

Rank deficiency of the feature map of Kar and Karnick (2012) was later demonstrated by Hamid et al. (2014). This meant that most of the weight components belonging to the model parameters of learning algorithms operating in this new feature space ($\mathbb{R}^m$) tend to be nearly zero. In order to address the issues of improving learning efficiency while achieving better kernel approximation accuracy, Hamid et al. (2014) proposed Compact Random FeaTure Maps (CRAFTMaps). Their idea was to use an existing dot product kernel map

for "up-projecting" (a non-linear mapping from $\mathbb{R}^d$ to $\mathbb{R}^m$) and subsequently "down-projecting" (a linear Johnson Lindenstrauss (JL) based mapping from $\mathbb{R}^m$ to $\mathbb{R}^{m'}$), where $m' < m$. In fact, (Hamid et al. 2014) experimentally demonstrated the benefit of using CRAFTMaps in improving the learning accuracy of Random Maclaurin and Tensor Sketching.

Pennington et al. (2015) proved that such polynomial kernel functions, although shift invariant, are not positive definite. Consequently, Bochner's theorem (Theorem 1) cannot be applied in this case. As a workaround, Pennington et al. (2015) proposed approximating the kernel function $K(\mathbf{x}, \mathbf{y})$ with a positive definite surrogate function $\hat{K}(\mathbf{x}, \mathbf{y})$, so as to make the application of Bochner's theorem possible. In this way a RFF formulation for the polynomial kernel can be obtained. $\hat{K}(\mathbf{x}, \mathbf{y})$ is constructed as the sum of N Gaussians, which is made possible by a theorem by Schoenberg (1938). The main approximation, namely that of finding $\hat{K}(\mathbf{x}, \mathbf{y})$ produces its inverse Fourier transform, $\hat{K}(\omega)$, from which we can obtain $p(\omega)$. This distribution can be used to sample the weight vectors, ω_i of RFF. Their method is referred to as Spherical Random Features (SRF). The SRF map does not benefit from the use of CRAFTMap because it exhibits a lesser degree of rank deficit than the other two approaches (Pennington et al. 2015).

4.1 Entry-wise guarantees

The Random Maclaurin map of Eq. 28 admits the following guarantee.

Theorem 15 (Kar and Karnick 2012) $\mathbb{P}[\sup_{\mathbf{x},\mathbf{y}\in\mathcal{X}} |\langle \mathbf{Z}(x), \mathbf{Z}(y)\rangle - K(\mathbf{x}, \mathbf{y})| > \epsilon] \leq 2\left(\frac{32RL}{\epsilon}\right)^{2d} e^{-\frac{m\epsilon^2}{8C^2}}$, *where* $C = pf(pR^2)$, $L = f'(R^2) + p^2R\sqrt{d}f'(pR^2)$ *and* $p > 1$.

A crucial assumption can be imposed on the input data, namely that it is l_2 normalized, that is for all $\mathbf{x} \in \mathbb{R}^d$, $\|\mathbf{x}\|_2 = 1$. This consequently affects the behavior of the polynomial kernel function.

$$\begin{aligned} K(\mathbf{x}, \mathbf{y}) &= (a + \langle \mathbf{x}, \mathbf{y}\rangle)^p \\ &= (a + 2 - 2\langle \mathbf{x}, \mathbf{y}\rangle)^p \end{aligned}$$

Moreover, this kernel can be written as a shift invariant kernel.

$$\begin{aligned} K(\mathbf{x}, \mathbf{y}) &= \left(1 - \frac{\|\mathbf{x} - \mathbf{y}\|^2}{a^2}\right)^p \\ &= \alpha(q + \langle \mathbf{x}, \mathbf{y}\rangle)^p \end{aligned}$$

where $\alpha = \left(\frac{2}{a^2}\right)^p$ and $q = \frac{a^2}{2} - 1$, $\|\mathbf{x} - \mathbf{y}\|_2^2 = \|\mathbf{x}\|^2 + \|\mathbf{y}\|^2 - 2\langle \mathbf{x}, \mathbf{y}\rangle$. The error bound obtained using the SRF method is $O(p^{-2.5} + 1/m)$.

Theorem 16 (Pennington et al. 2015) *The error of the SRF map is upper bounded by the error attributed to Monte-Carlo sampling, which is* $O(1/m)$ *and the error attributed to the surrogate kernel function, which is* $p^{-2.5}$.

The Tensor Sketching approach of Pham and Pagh (2013) admits the following tighter bound (Theorem 17) when compared to Theorem 15.

Theorem 17 (Pham and Pagh 2013) *For* $X = \frac{1}{m}\sum_{i=1}^{m} X_i$, *where* $X_i = \langle \mathbf{Z}(x), \mathbf{Z}(y) \rangle$, $R^{2p} \leq X_i \leq R^{2p}$, $R \in \mathbb{R}^+$ *and any* $\epsilon > 0$,

$$Pr[|X - \mathbb{E}[X]| \geq \epsilon] \leq 2e^{\frac{-m\epsilon^2}{2R^{4p}}}$$

The following entry wise bounds that are independent of d were proved by Hamid et al. (2014) for their CRAFTMaps method (Theorem 18).

Theorem 18 (Hamid et al. 2014) *Given any two unit vectors* $\mathbf{x}$ *and* $\mathbf{y}$ *of d dimensions, applying a random feature mapping of* Kar and Karnick (2012) *(*$\mathbf{Z}$: $\mathbb{R}^d \to \mathbb{R}^m$*) followed by a JL map (*$\mathbb{R}^m \to \mathbb{R}^{m'}$*) to these vectors to obtain* $\mathbf{Z}'$, *the following bound is obtained.* $\langle \mathbf{x}, \mathbf{y} \rangle^p - \langle \mathbf{Z}'(\mathbf{x}), \mathbf{Z}'(\mathbf{y}) \rangle \leq m^{-1/2}(2^{p+1}\log(n)^{p+1}) + {m'}^{-1/2}(\log(n)^{1/2})$.

Besides, this improved error bound, better classification results were also obtained using CRAFTMaps.

5 Summary and discussion

In this section, summary of the advancements made and important questions in regard to shift invariant kernel approximation are discussed.

The summary of the major algorithmic developments in shift-invariant kernel approximation are given in Table 1. In this table, the mapping time is reported for any vector of d dimensions. It is clear from this table that most of the works with the exception of RFF do not have learning guarantees. The fastest kernel map in shift invariant category include quadrature based methods (GQ and BQ) and structured matrices based (Fastfood, SORF, ROM). In the dot product category, CRAFTMaps is the most computationally efficient kernel map.

Table 2 summarizes the entry-wise bounds for the approximations and their corresponding bound on m. It can be observed that improvement in the error bounds have been made in recent works. However, tight bounds on the sample complexity (m) have not been provided yet. Besides providing kernel matrix approximation and entry-wise guarantees, bounds with respect to the learning task it is applied in are also extremely useful. Data-obliviousness property ensures that the kernel mapping can be applied in an incremental manner, which is especially useful in streaming applications. But, recently research has also been done on data-dependent kernel maps. This idea has been explored with the aim of improving learning accuracy and reducing the number of samples (m) needed to achieve acceptable accuracy, which is application dependent.

Next, some important questions pertaining to the recent results are discussed.

1. *Does changing the sampling distribution make a difference?* The original idea of RFF sampled from the Fourier transform of the shift-invariant kernel (Gaussian kernel). Due to this construction of RFF, a large number of samples are required in order to achieve good kernel approximation accuracy (Huang et al. 2014; Lopez-Paz et al. 2014).

Table 1 Main algorithmic development in kernel approximation

Type of kernel	Work	Mapping time	Data oblivious?	Approximation guarantees	Learning guarantees
Shift invariant	RFF (Rahimi et al. 2007)	$O(md)$	✓	Entry-wise	✓
	Fastfood (Le et al. 2013)	$O(m\log(d))$	✓	Entry-wise	✗
	ORF (Felix et al. 2016)	$O(md)$	✓	Entry-wise	✗
	SORF (Felix et al. 2016)	$O(m\log(d))$	✓	Entry-wise	✗
	ROM (Choromanski et al. 2017)	$O(m\log(d))$	✓	Entry-wise	✗
	QMC (Avron et al. 2016)	$O(md)$	✓	Kernel approximation based on discrepancy measure	✗
	Gaussian quadrature (Dao et al. 2017)	$O(m\log(d))$	✓	Entry-wise	✗
	BQ (Munkhoeva et al. 2018)	$O(m\log(d))$	✓	Entry-wise	✗
Dot product	Random Maclaurin (Kar and Karnick 2012)	$O(md)$	✓	Entry-wise	✗
	Tensor sketching (Pham and Pagh 2013)	$O(p(d+m\log(m)))$	✓	Entry-wise	✗
	CRAFTMaps (Hamid et al. 2014)	$O(m\log(d)+m\log(m))$	✓	Entry-wise	✗
	SRF (Pennington et al. 2015)	$O(md)$	✓	Entry-wise	✗

Table 2 Entry-wise bounds and sample sizes

Type of kernel	Reference	Entry-wise error	Bound for m
Shift invariant	Lemma 3	$O\left(e^{-m\epsilon^2/2}\right)$	$\Omega(d/\epsilon^{-2})$
	Theorem 2	$O\left(e^{-m\epsilon^2/4(d+2)}\right)$	
	Theorem 3	$O\left(e^{-m\epsilon^2/8(d+2)\alpha}\right)$	
	Theorem 4	$O\left(e^{-m\epsilon^2/32(d+1)\alpha'}\right)$	
	Theorem 6	$O\left(\sqrt{\log\lvert M\rvert/m}\right)$	$O(\epsilon^{-2}\log\lvert M\rvert)$
	Theorem 7	$O(M^L)$	$O(1/\epsilon^{\gamma})$
	Theorem 8	$O\left(e^{-m\epsilon^2/8M^2(d+1)}\right)$	$\Omega(d/\epsilon^{-2})$
Dot product	Theorem 15	$O(\epsilon^{-2d}e^{-m\epsilon^2/C^2})$	$\Omega(\epsilon^{-2})$
	Theorem 17	$O(e^{-m\epsilon^{2/p}})$	
	Theorem 18	$O(m^{-1/2}2^{p+1}+m'^{-1/2})$	
	Theorem 16	$O(p^{-2.5}+1/m)$	

- *Structured matrices* By using structured matrices, Le et al. (2013) achieved faster kernel mapping at the cost of slightly worse approximation error. Such matrices have been used in many different applications such as to speed up kernel matrix approximation, dimensionality reduction (Ailon and Chazelle 2006; Halko et al. 2011), compressed sensing (Nelson et al. 2014) etc. Felix et al. (2016) used such matrices to improve approximation error as well as achieve lesser mapping time. Experiments with various kinds of structured matrices performed by Choromanski and Sindhwani (2016) measuring the entry-wise error and kernel matrix approximation error indicate that a using structured matrices (ROM) incurs the least kernel matrix and entry-wise error.
- *Quasi-random sampling approach* The idea of using low-discrepancy sequences (Avron et al. 2016) provides an error convergence rate of $(O(\log(m)^d/m))$. This upper bound is not useful when d is large due to the $\log(m)^d$ term in the numerator. In fact, this method is effective only when d is not too large.
- *Quadrature based* The Gaussian quadrature based approach of Dao et al. (2017) appears to have better approximation guarantees when compared to RFF. The follow-up work by Munkhoeva et al. (2018) that combines quadrature rules and structured matrices also support this claim. In fact, their method established the connection between ORF and spherical-radial quadrature rules and demonstrated good classification/regression performance of their approach on some datasets.
- *Ridge-leverage score based* Avron et al. (2017) proposed sampling from an approximation to a ridge leverage function. Their motivation was to provide tight upper bounds on m in order to achieve good kernel approximation bounds. Since there does not exist an efficient way to sample from the ridge leverage score function distribution, an "approximation" to it is employed. Although good kernel approximation bounds are obtained, this is done at the cost of having a very loose upper bound on m (exponential in d). This is not useful in practical where d is usually very large.

Thus, changing the sampling distribution does help in achieving lower error of approximation, reducing kernel mapping time and better learning accuracy.

2. *How big should m be in practice?* Ideally we would want m to be much smaller than d and n in order to ensure feasible computational costs. Depending on the kind of approxi-

mation, m can range from a very small quantity to extremely large values (exponential in d). The right value of m can vary from one application to another. In fact, works such as those of Lin et al. (2014), Hu et al. (2015) gave bounds for m in the context of online learning and showed that the value of m is often tied to the learning problem. The bounds proposed so far are not tight and depend on a function of the regularization parameter.

3. *What are the merits of adding dependency over data-oblivious approach?* While data obliviousness has advantages over the data dependent approach, the latter was shown by Sinha and Duchi (2016), Shahrampour et al. (2017) to require less number of random features for achieving the learning goal. However, both these works have expensive computational overheads for data with large d.
4. *Does good kernel matrix approximation imply good learning performance?* Most of the previous works focus on proving that their methods are good kernel (matrix/entry-wise) approximations, however the question regarding whether a good kernel matrix approximation automatically guarantees good learning performance remains. In fact, previous research (Schleif and Tino 2017; Schleif et al. 2020) indicates that constructing the learning boundaries accurately is more important than constructing accurate kernel matrices.

   In Sect. 6.1 we investigate this problem with some experimental results.

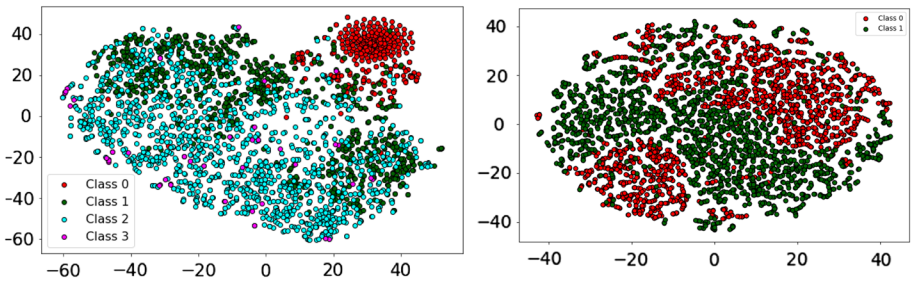
6 Experiments and applications of kernel approximation

6.1 Experiments

In this section, the following experiments are performed.

1. Study the effect of m (kernel mapping parameter) on kernel matrix approximation accuracy.
2. Study the performance of kernel approximation on classification tasks.

Dataset description: The datasets used for the experiments include Parkinson's Progressive Markers Initiative (PPMI) (Marek et al. 2011) dataset and COD-RNA (Uzilov et al. 2006). The PPMI dataset contains the motor assessments of patients with 4379



(a) PPMI dataset **(b)** COD-RNA dataset

Fig. 3 t-SNE plots of the datasets

instances and 34 attributes, for other details please refer to the work of Francis and Raimond (2016). The COD-RNA dataset contains 488,565 instances and 8 attributes. The t-Distributed Stochastic Neighbor Embedding (t-SNE) algorithm plots of the two datasets used in the experiments are shown in Fig. 3. This plot between the first two features of the t-SNE transformed data shows that separating the instances of the different classes cannot be done using a linear boundary. This is an indication that kernel based techniques are more appropriate for these datasets. We have not included large datasets (n and/or d is large) because computing the kernel matrix would be computationally and storage-wise infeasible.

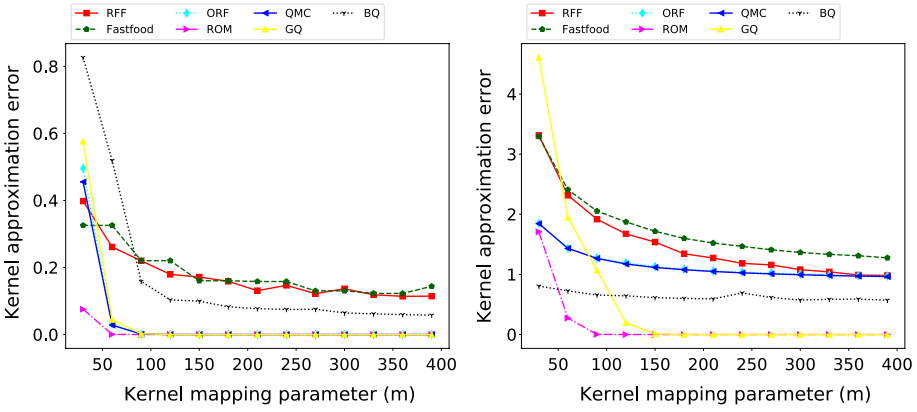
Algorithms being compared: In the experiments, the following RBF kernel approximations are considered: **RFF** (Rahimi et al. 2007), **Fastfood** (Le et al. 2013), **ORF** (Felix et al. 2016), **ROM** (Choromanski et al. 2017), **QMC** (Avron et al. 2016) and **BQ** (Munkhoeva et al. 2018).

We have used smaller values of m for all our experiments in contrast to previous works. This is because we are interested in studying the error and accuracy for small values of m. For timing results of all the above algorithms refer to (Munkhoeva et al. 2018). Comprehensive experiments studying the kernel approximation accuracy and learning accuracy of the dot product kernel approximation methods like Random Maclaurin, Tensor Sketch, CRAFTMaps and SRF have already been provided in the work of Pennington et al. (2015). Therefore, we do not include such experiments with approximations to the dot-product kernel in order to avoid repeating the results.

6.1.1 Kernel matrix approximation accuracy

The kernel approximation accuracy is computed as follows (Eq. 29).

$$err = \frac{\|\mathbf{K}_{exact} - \mathbf{K}_{approx}\|_F}{\|\mathbf{K}_{exact}\|_F} \tag{29}$$



(a) PPMI dataset **(b)** COD-RNA dataset

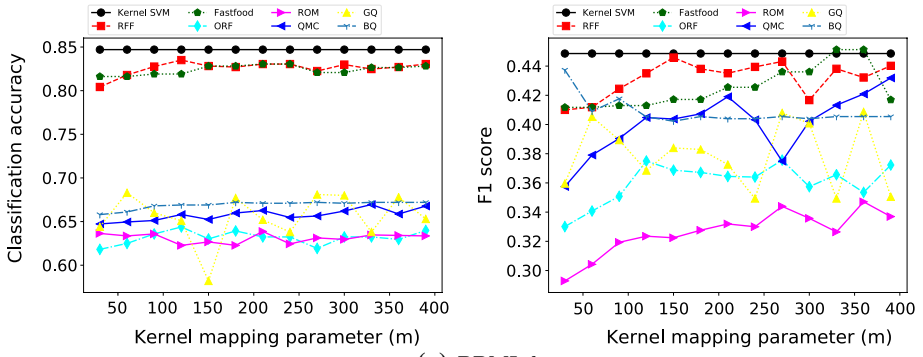
Fig. 4 Variation of kernel approximation error with kernel mapping parameter

Figure 4 shows the plots between kernel approximation accuracy and kernel mapping parameter (m). The lowest error is incurred by GQ, QMC and ROM for $m \geq 100$, but for lower values of m ($m < 50$), ROM obtain the lowest error. For the PPMI dataset in practice, we would want m to be quite smaller than d, so in the case of PPMI dataset, ROM method is the most accurate in terms of kernel approximation. For the COD-RNA dataset, BQ is the most accurate for small values of m ($m < 50$) and for larger values of m, ROM and GQ appear to have the lowest error.

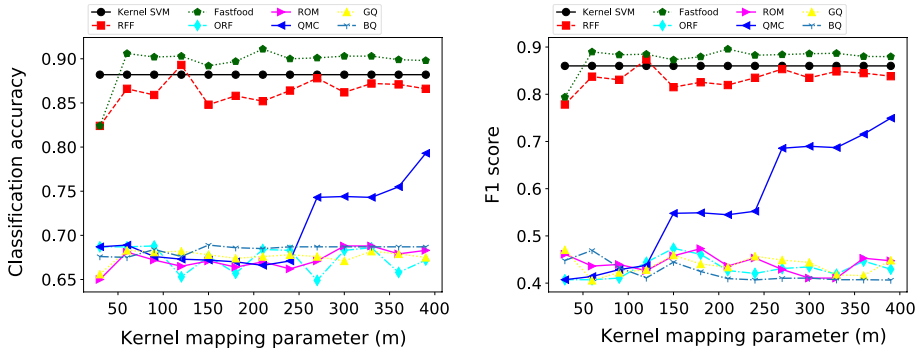
6.1.2 Learning performance

The learning performance is measured using two metrics, accuracy and F1-score. Since there is significant imbalance in the number of instances of each class in the datasets, accuracy alone is not a good measure of the classifier performance. F1-score is a better measure of classifier performance in the presence of class imbalance. The classification algorithm used is linear Support Vector Machine (SVM). We also include **Kernel SVM** algorithm in this study in order to compare its results with those obtained through approximate kernel based methods. The Kernel SVM algorithm simply uses the RBF kernel followed by SVM classifer.

The results are shown in Fig. 5. The results of Kernel SVM remains a constant in all plots because m is not a parameter of this classifier. The plots show that only RFF



(a) PPMI dataset



(b) COD-RNA dataset

Fig. 5 Variation of classification accuracy and F1-score with kernel mapping parameter

and Fastfood try to match the classification accuracy of Kernel SVM. In the COD-RNA dataset (Fig. 5b), Fastfood + linear SVM achieves better accuracy and F1-score than Kernel SVM. For the PPMI dataset (Fig. 5a) BQ algorithm + linear SVM achieves comparable F1-score with RFF and Fastfood. The ROM algorithm + linear SVM also tries to match the F1-score of Fastfood and RFF for PPMI dataset for $m \geq 150$. ROM, GQ and ORF do not fare well in both datasets.

6.1.3 Summary

The results obtained in Sect. 6.1.2 indicate that achieving good kernel approximation accuracy does not imply good learning performance. In fact, the results indicate that the accurate kernel approximations do not perform well on classification tasks on the datasets used in the experiments.

Next, let us consider the value of the kernel mapping parameter m. In practical applications, we ideally want m to be much smaller than d. Therefore it does not make sense to use values of $m = 200$ for a dataset such as PPMI with $d = 34$ or COD-RNA with $d = 8$. Based on this observation, we argue that RFF and Fastfood obtain the best classification performance, even though they incur higher kernel approximation errors than other methods.

6.2 Applications

In this section, some of the important applications of the schemes discussed earlier are described. Table 3 lists the applications of shift-invariant kernel approximation. RFF has been used in the practical learning of deep Gaussian processes, scalable semi-supervised kernel spectral learning (Mehrkanoon and Suykens 2016), kernel deep convex network (Huang et al. 2013) and acoustic modeling in speech recognition (May et al. 2017). An empirical comparison of some KPCA approximation (a direct application of RFF) schemes in the context of classification was done by Francis and Raimond (2017). RFF and its variants have also been used for the application of streaming anomaly detection. Francis and Raimond (2020) proposed an accurate explicit kernel map called Explicit Cosine Map (ECM) that was demonstrated to be more accurate than other approximate kernel maps in the application of anomaly detection.

6.3 Kernel clustering

Kernel clustering algorithms aim to discover arbitrary shaped clusters from the given data. If the dataset size is $n \times d$, then $O(nd)$ space is required to store the whole dataset in memory. When n (number of data instances) is extremely large, storage becomes an issue. There are various ways to deal with this issue. In particular, streaming batch or subset based processing is a widely adopted approach, wherein the data are partitioned (at the source) into various batches and the processing is done on these batches. The task of clustering such data is often done using incremental cluatering algorithms. Such algorithms are built from popular algorithms such as k-means and k-medoids. But such traditional kernel based clustering algorithms suffer from poor scalability, and hence are unsuitable for large datasets.

Table 3 Applications of kernel approximation

Work	Kernel approximation	Area	Theoretical results	Practical results
Chitta et al. (2012)	RFF	K-means clustering	✓	✓
Huang et al. (2013)	RFF	KRR using K-DCN	✗	✓
Huang et al. (2014)	RFF	KRR, Speech recognition	✗	✓
Lopez-Paz et al. (2014)	RFF	KPCA	✓	✓
Lin et al. (2014)	RFF	Online learning	✓	✓
Sutherland and Schneider (2015)	RFF	KRR, SVM, Maximum mean discrepancy	✓	✗
Ghashami et al. (2016)	RFF	KPCA	✓	✓
Cutajar et al. (2016)	RFF	DGP	✓	✓
Lu et al. (2016)	RFF	Online learning	✓	✓
Avron et al. (2017)	Ridge leverage score based-RFF	KRR	✓	✓
Francis and Raimond (2018b)	RFF	Streaming anomaly detection	✓	✓
He et al. (2018)	Fastfood	Spectral clustering	✗	✓
Carratino et al. (2018)	RFF	Online learning	✓	✓
Francis and Raimond (2020)	RFF, Fastfood, ORF, SORF	Streaming anomaly detection	✗	✓

Possible solutions have been proposed in the form of approximate kernel based clustering using RFF and Fastfood. A straightforward adaptation of RFF into k-means algorithm results in the following issues.

- *High computational cost:* this is due to the fact that approximate kernel clustering error decreases as kernel mapping parameter, m increases (Theorem 2 of Chitta et al. 2012). Thus in order to achieve good kernel clustering accuracy, k-means has to be run over of a high dimensional space of $\mathbb{R}^m$, where m is large.
- *Increased storage costs:* The space needed to store the approximate kernel matrix is $O(m^2)$, which can be large according to the previous point.

RFF has been successfully incorporated in a scalable kernel clustering algorithm (Chitta et al. 2012). Their approach to solving the first issue mentioned above involves imposing a constraint on the cluster centers by ensuring they lie in the subspace spanned by the top C eigenvectors of the kernel matrix. But this approach is computationally inefficient because it requires the eigen decomposition of the $n \times n$ kernel matrix. This issue is overcome by using the approximate kernel matrix $\mathbf{K}_{approx}$ obtained by applying RFF. By computing the top C singular values and vectors of $\mathbf{K}_{approx}$, a new vector space is obtained over which k-means is run. Their experimental results indicate that the new algorithm performs better than other popular kernel clustering algorithms.

Kernel k-means is a special case of the general spectral clustering approach. The latter relies on the eigen decomposition of the kernel matrix, which can be very computational expensive. He et al. (2018) tackled the issue of spectral clustering on large datasets using an incremental kernel based clustering using Fastfood. Their method used a streaming and incremental approach to cluster the given data points. In their streaming approach, a single data point is applied the Fastfood kernel map and the subsequently obtained feature vector is used to incrementally compute the eigenvectors of the kernel matrix. They demonstrated improvements in accuracy and speedup over previous clustering methods.

The second issue regarding the storage costs of storing the kernel matrix becomes important when m is very large. This again brings out the question of what the optimal value of m is. The answer is tied to the kind of application we are dealing with. In kernel based clustering, accuracy of clustering determines storage and computational needs. As in the case of any approximate solution, there is a trade-off between accuracy and computational and space requirements.

6.4 Deep learning

Deep Gaussian Process (DGP) (Damianou and Lawrence 2013) is a non-parametric deep learning technique that is composed of Gaussian Process (GP) mappings which helps to automatically learn hierarchical features from the data. This technique offers advantages such as ability to learn from small datasets and ease of determining the optimal number of hidden layers in the network. This method however suffers from the following practical issues.

- Computationally expensive: DGP requires the computation of the probability of the labels given the input and other parameters, $Pr(\mathbf{y}|\mathbf{X}, \boldsymbol{\Theta})$, which is very expensive. Many useful likelihood functions are analytically intractable. Besides, some of the algebraic calculations involves are of the order of $O(n^3)$.

By using RFF feature maps at each layer, (Cutajar et al. 2016) proposed a solution to the problem described above. This method also provides a way of providing regularization by imposing a low-rank structure in the weights of the network. This new approach also leads to better performance in classification and regression tasks, when compared to other state-of-the-art DGP models (Cutajar et al. 2016).

Kernel Deep Convex Network (K-DCN) is a deep learning technique that uses the kernel trick to make the Deep Convex Network (DCN) more powerful. The use of traditional Gaussian kernel becomes infeasible when the dataset is large. In order to overcome the scalability issues of K-DCN, its approximate version (Huang et al. 2013) was proposed by incorporating RFF in the hidden layers. Their method was demonstrated to achieve scalability and good accuracy in learning tasks.

By incorporating RFF into the deep learning architectures, the kernel mapping parameter m becomes a tuneable hyperparameter and its optimal values can be learned automatically from the data.

6.5 KRR

A notable result in the task of KRR was obtained by Huang et al. (2014). By combining RFF and convex optimization, matching or better performance than DNNs was obtained on the TIMIT dataset. A scalable solver consisting of block coordinated descent algorithm is applied on the implicitly defined matrix $\mathbf{Z}$ (obtained by applying RFF on the data). This matrix $\mathbf{Z}$ is distributed across machines, and the block coordinate descent algorithm is run in parallel on all these machines. This distribution across machines is possible because of the data obliviousness of RFF, in which a random feature computation can be done independent of others. This parallel solver is used along with an ensemble approach that results in a powerful model capable of matching the performance of DNNs.

6.6 Online learning

In the area of online kernel learnining, where the aim is to learn kernel based models for prediction from data that arrive sequentially. Some of the key challenges in this area include the following.

- *Computational inefficiency:* some of the efficient algorithms in the area of online kernel learning despite being accurate are very computationally expensive.
- *Large communication costs:* this issue arises due to the growing size of support vectors.

Solutions to the above two problems were provided by Lu et al. (2016) by incoporating kernel function approximation, RFF into the online learning method. The data oblivious nature of RFF makes it an ideal kernel mapping technique for online learning. Data are first transformed to the new feature space using RFF, and subsequently an online learning task is solved using an algorithm such as gradient descent. It was demonstrated that such an approach produces more accurate results than previous online kernel learning approaches. Learning with Stochastic Gradient Descent (SGD) and random features was demonstrated

by Carratino et al. (2018). They showed that by using $\sqrt{n}$ random features, certain optimal statistical properties were achieved by this learning method.

Various strategies for kernel based online distributed learning have been proposed in the past. However, due to the nature of distributed computations, standard kernel functions are not applicable because the amount of messages (also called communication cost) transferred between nodes in the communication network will be prohibitively large. A solution to this problem was given by Bouboulis et al. (2017) in the form of an online distributed learning protocol that uses RFF.

7 Open problems and possible future directions

The data obliviousness of explicit feature maps offer the following advantages.

- Can be applied in a streaming manner.
- Data independent guarantees for number of samples and time taken can be obtained.

On the downside, the following are the consequences of having data obliviousness.

- Large number of samples are needed to obtain good kernel approximation.
- Good performance in learning problems cannot be guaranteed due to the varying nature of data.

Some of the key problems that could be addressed in the future are as follows.

- *Nature of the data:* A generalized approach for feature mapping needs to takes into account the nature of data, and the closely tied problem of learning for accomplishing a particular task. The relation between performance of RFF and nature of the data is yet to be explored extensively. A possible avenue for exploration in this regard includes data dependent kernel learning. The idea of selecting the best kernel (or kernel parameters) by solving an optimization problem on the data is powerful idea that can be explored further.
- *Number of random features (m) required in practice:* Upper bounds on the number of random features (m) required with respect to kernel approximation accuracy and learning tasks have been provided by previous works (Rahimi et al. 2007; Lin et al. 2014; Avron et al. 2017). But, they are loose bounds and do not provide insight into the value of m to be used in practice. The optimal value of m varies from application to application.
- *Learning guarantees:* Although guarantees with respect to learning tasks have been developed for RFF, many of its variants such as the structured matrix based techniques like ORF, SORF, ROM, quadrature based methods are without such guarantees. Our experiments in Sect. 6.1 indicate that RFF and its variant Fastfood perform well on learning tasks despite having higher error of kernel approximation. The results indicate that low kernel approximation error does not imply good learning accuracy. This observation has also been made by other research works such as Schleif and Tino (2017), Schleif et al. (2020). From a practical viewpoint, the problem of achieving good learning performance is dependent on (1) the choice of the learning algorithm (2) the choice of good hyperparameters for the learning algo-

rithm (3) determining whether kernel based methods are appropriate for the given dataset and task and (4) the choice of appropriate kernel. The points (1) and (2) are often considered in the literature of learning algorithms. However, the last two points (3) and (4) are given less priority.

- *Unsupervised data-dependent kernel learning with guarantees:* The preliminary ideas for learning the best kernel in an unsupervised setting has been described in the work of Damodaran et al. (2017). This idea has the potential for improving the accuracy of unsupervised learning tasks that require kernels. In this regard, it would be interesting to develop guarantees with respect to its performance.

8 Conclusion

Recent theoretical advancements in the area of kernel function approximation were reviewed in this work. Most of these works focused on improving the running time and performance, providing improved analysis, and demonstrating the usefulness in real-world learning tasks. The major breakthrough in the area of kernel function approximation was obtained by Rahimi et al. (2007) in the form of RFF. In the recent years, many improvements of RFF and advancements in the theory of kernel function approximation were proposed. Among these are the methods that provided major run-time improvements. The best running time improvement was obtained by using structured matrices to approximate the kernel map. Sampling custom random features for improving the performance of learning algorithms has also been explored in the past. The idea of data dependent kernel learning is a powerful idea that has the potential to produce further breakthroughs in the area of kernel function approximation. The various kernel function approximation methods have been succesfully used in areas such as deep learning, kernel clustering and online learning. Although RFF and Fastfood are the most popular, other computationally efficient strategies such as structured matrices based and quadrature based methods can also be incorporated into such applications. Our experiments on kernel approximation error and classification accuracy indicate that achieving low values of the former does not imply good learning performance. The open problems described in this work also paves the way for future research in this area.

Acknowledgements This publication is an outcome of the R&D work undertaken project under the Visvesvaraya Ph.D. Scheme of Ministry of Electronics and Information Technology, Government of India, being implemented by Digital India Corporation.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

References

Ailon N, Chazelle B (2006) Approximate nearest neighbors and the fast Johnson–Lindenstrauss transform. In: Proceedings of the thirty-eighth annual ACM symposium on theory of computing, pp 557–563

Aronszajn N (1950) Theory of reproducing kernels. Trans Am Math Soc 68(3):337–404

Avron H, Sindhwani V, Yang J, Mahoney MW (2016) Quasi-Monte Carlo feature maps for shift-invariant kernels. J Mach Learn Res 17(1):4096–4133

Avron H, Kapralov M, Musco C, Musco C, Velingker A, Zandieh A (2017) Random Fourier features for kernel ridge regression: approximation bounds and statistical guarantees. In: Proceedings of the 34th international conference on machine learning, PMLR, Proceedings of machine learning research, vol 70, pp 253–262, http://proceedings.mlr.press/v70/avron17a.html
Bach F (2013) Sharp analysis of low-rank kernel matrix approximations. In: Conference on learning theory, pp 185–209
Bach F (2017) On the equivalence between kernel quadrature rules and random feature expansions. J Mach Learn Res 18(21):1–38
Bouboulis P, Chouvardas S, Theodoridis S (2017) Online distributed learning over networks in RKH spaces using random Fourier features. IEEE Trans Signal Process 66(7):1920–1932
Carratino L, Rudi A, Rosasco L (2018) Learning with SGD and random features. In: Advances in neural information processing systems, pp 10192–10203
Cesa-Bianchi N, Conconi A, Gentile C (2004) On the generalization ability of on-line learning algorithms. IEEE Trans Inf Theory 50(9):2050–2057
Chang PC, Wu JL (2015) A critical feature extraction by kernel PCA in stock trading model. Soft Comput Fusion Found Methodol Appl 19(5):1393–1408
Charikar M, Chen K, Farach-Colton M (2002) Finding frequent items in data streams. In: International colloquium on automata, languages, and programming. Springer, pp 693–703
Chaudhuri K, Monteleoni C, Sarwate AD (2011) Differentially private empirical risk minimization. J Mach Learn Res 12(Mar):1069–1109
Chitta R, Jin R, Jain AK (2012) Efficient kernel clustering using random Fourier features. In: 2012 IEEE 12th international conference on data mining (ICDM). IEEE, pp 161–170
Choromanski K, Sindhwani V (2016) Recycling randomness with structure for sublinear time kernel expansions. In: Proceedings of the 33rd international conference on international conference on machine learning, vol 48, JMLR.org, pp 2502–2510
Choromanski KM, Rowland M, Weller A (2017) The unreasonable effectiveness of structured random orthogonal embeddings. In: Advances in neural information processing systems, pp 219–228
Cohen MB, Musco C, Musco C (2015) Ridge leverage scores for low-rank approximation. arXiv preprint arXiv:1511.07263 6
Cutajar K, Bonilla EV, Michiardi P, Filippone M (2016) Practical learning of deep Gaussian processes via random Fourier features. arXiv preprint arXiv:1610.04386
Damianou A, Lawrence N (2013) Deep Gaussian processes. In: Artificial intelligence and statistics, pp 207–215
Damodaran BB, Courty N, Gosselin PH (2017) Data dependent kernel approximation using pseudo random Fourier features. arXiv preprint arXiv:1711.09783
Dao T, De Sa CM, Ré C (2017) Gaussian quadrature for kernel features. In: Advances in neural information processing systems, pp 6109–6119
Drineas P, Mahoney MW (2005) On the Nyström method for approximating a gram matrix for improved kernel-based learning. J Mach Learn Res 6(Dec):2153–2175
Felix XY, Suresh AT, Choromanski KM, Holtmann-Rice DN, Kumar S (2016) Orthogonal random features. In: Advances in neural information processing systems, pp 1975–1983
Francis DP, Raimond K (2016) Comparison of machine learning techniques for the identification of the stages of Parkinson's disease. In: Senthilkumar M, Ramasamy V, Sheen S, Veeramani C, Bonato A, Batten L (eds) Computational intelligence, cyber security and computational models. Springer, Singapore, pp 247–259. https://doi.org/10.1007/978-981-10-0251-9_25
Francis DP, Raimond K (2017) Empirical evaluation of kernel PCA approximation methods in classification tasks. arXiv preprint arXiv:1712.04196
Francis DP, Raimond K (2018a) An improvement of the parameterized frequent directions algorithm. Data Min Knowl Discov 32(2):453–482
Francis DP, Raimond K (2018b) A random Fourier features based streaming algorithm for anomaly detection in large datasets. In: Advances in big data and cloud computing. Springer, pp 209–217
Francis DP, Raimond K (2020) A fast and accurate explicit kernel map. Appl Intell 50(3):647–662
Ghashami M, Desai A, Phillips JM (2014) Improved practical matrix sketching with guarantees. In: European symposium on algorithms. Springer, pp 467–479
Ghashami M, Perry DJ, Phillips J (2016) Streaming kernel principal component analysis. In: Proceedings of the 19th international conference on artificial intelligence and statistics, pp 1365–1374
Halko N, Martinsson PG, Tropp JA (2011) Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. SIAM Rev 53(2):217–288
Hamid R, Xiao Y, Gittens A, DeCoste D (2014) Compact random feature maps. In: International conference on machine learning, pp 19–27

He L, Li Y, Zhang X, Chen C, Zhu L, Leng C (2018) Incremental spectral clustering via fastfood features and its application to stream image segmentation. Symmetry 10(7):272
Hu Z, Lin M, Zhang C (2015) Dependent online kernel learning with constant number of random Fourier features. IEEE Trans Neural Netw Learn Syst 26(10):2464–2476
Huang PS, Deng L, Hasegawa-Johnson M, He X (2013) Random features for kernel deep convex network. In: 2013 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE, pp 3143–3147
Huang PS, Avron H, Sainath TN, Sindhwani V, Ramabhadran B (2014) Kernel methods match deep neural networks on timit. In: ICASSP, pp 205–209
Kar P, Karnick H (2012) Random feature maps for dot product kernels. AISTATS 22:583–591
Kulis B, Grauman K (2012) Kernelized locality-sensitive hashing. IEEE Trans Pattern Anal Mach Intell 34(6):1092–1104
Le Q, Sarlós T, Smola A (2013) Fastfood-approximating kernel expansions in loglinear time. In: Proceedings of the international conference on machine learning
Lee JM, Yoo C, Choi SW, Vanrolleghem PA, Lee IB (2004) Nonlinear process monitoring using kernel principal component analysis. Chem Eng Sci 59(1):223–234
Liberty E (2013) Simple and deterministic matrix sketching. In: Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, pp 581–588
Lin M, Weng S, Zhang C (2014) On the sample complexity of random Fourier features for online learning: How many random Fourier features do we need? ACM Trans Knowl Discov Data (TKDD) 8(3):13
Lopez-Paz D, Sra S, Smola AJ, Ghahramani Z, Schölkopf B (2014) Randomized nonlinear component analysis. In: ICML, pp 1359–1367
Lu J, Hoi SC, Wang J, Zhao P, Liu ZY (2016) Large scale online kernel learning. J Mach Learn Res 17(1):1613–1655
Mackey L, Jordan MI, Chen RY, Farrell B, Tropp JA et al (2014) Matrix concentration inequalities via the method of exchangeable pairs. Ann Probab 42(3):906–945
Mahoney MW (2011) Randomized algorithms for matrices and data. Found Trends® Mach Learn 3(2):123–224
Marek K, Jennings D, Lasch S, Siderowf A, Tanner C, Simuni T, Coffey C, Kieburtz K, Flagg E, Chowdhury S et al (2011) The Parkinson progression marker initiative (PPMI). Prog Neurobiol 95(4):629–635
May A, Garakani AB, Lu Z, Guo D, Liu K, Bellet A, Fan L, Collins M, Hsu D, Kingsbury B, et al (2017) Kernel approximation methods for speech recognition. arXiv preprint arXiv:1701.03577
Mehrkanoon S, Suykens JA (2016) Scalable semi-supervised kernel spectral learning using random Fourier features. In: 2016 IEEE symposium series on computational intelligence (SSCI). IEEE, pp 1–8
Munkhoeva M, Kapushev Y, Burnaev E, Oseledets I (2018) Quadrature-based features for kernel approximation. In: Advances in neural information processing systems, pp 9147–9156
Musco C, Musco C (2016) Provably useful kernel matrix approximation in linear time. arXiv preprint arXiv:1605.07583
Nelson J, Price E, Wootters M (2014) New constructions of RIP matrices with fast multiplication and fewer rows. In: Proceedings of the twenty-fifth annual ACM-SIAM symposium on discrete algorithms. SIAM, pp 1515–1528
Pennington J, Felix XY, Kumar S (2015) Spherical random features for polynomial kernels. In: Advances in neural information processing systems, pp 1846–1854
Pham N, Pagh R (2013) Fast and scalable polynomial kernels via explicit feature maps. In: Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, pp 239–247
Rahimi A, Recht B (2009) Weighted sums of random kitchen sinks: replacing minimization with randomization in learning. In: Advances in neural information processing systems, pp 1313–1320
Rahimi A, Recht B, et al. (2007) Random features for large-scale kernel machines. In: NIPS, vol 3, p 5
Rudi A, Rosasco L (2017) Generalization properties of learning with random features. In: Advances in neural information processing systems, pp 3218–3228
Schleif FM, Tino P (2017) Indefinite core vector machine. Pattern Recognit 71:187–195
Schleif FM, Raab C, Tino P (2020) Sparsification of core set models in non-metric supervised learning. Pattern Recognit Lett 129:1–7
Schoenberg IJ (1938) Metric spaces and completely monotone functions. Ann Math 39:811–841
Shahrampour S, Beirami A, Tarokh V (2017) On data-dependent random features for improved generalization in supervised learning. arXiv preprint arXiv:1712.07102
Shahrampour S, Beirami A, Tarokh V (2018) Supervised learning using data-dependent random features with application to seizure detection. In: 2018 IEEE conference on decision and control (CDC). IEEE, pp 1168–1173

Sinha A, Duchi JC (2016) Learning kernels with random features. In: Advances in neural information processing systems, pp 1298–1306
Sriperumbudur B, Szabó Z (2015) Optimal rates for random Fourier features. In: Advances in neural information processing systems, pp 1144–1152
Sutherland DJ, Schneider J (2015) On the error of random Fourier features. In: Proceedings of the thirty-first conference on uncertainty in artificial intelligence. AUAI Press, pp 862–871
Uzilov AV, Keegan JM, Mathews DH (2006) Detection of non-coding RNAs on the basis of predicted secondary structure formation free energy change. BMC Bioinform 7(1):143–173
Wang Q (2012) Kernel principal component analysis and its applications in face recognition and active shape models. arXiv preprint arXiv:1207.3538
Williams CK, Seeger M (2000) Using the Nyström method to speed up kernel machines. In: Proceedings of the 13th international conference on neural information processing systems. MIT press, pp 661–667