A Double-Variational Bayesian Framework in Random Fourier Features for Indefinite Kernels

Fanghui Liu, Xiaolin Huang, Lei Shi, Jie Yang, and Johan A.K. Suykens

***Abstract*—Random Fourier features (RFF) have been successfully employed to kernel approximation in large scale situations. The rationale behind RFF relies on Bochner's theorem, but the condition is too strict and excludes many widely-used kernels, e.g., dot-product kernels and indefinite kernels. In this paper, we present a unified RFF framework for indefinite kernel approximation in the Reproducing Kernel Kreĭn Spaces (RKKS). Besides, our model is also suited to approximate a dot-product kernel on the unit sphere, as it can be transformed into a shift-invariant but indefinite kernel. By the Kolmogorov decomposition scheme, an indefinite kernel in RKKS can be decomposed into the difference of two unknown positive definite (PD) kernels. The spectral distribution of each underlying PD kernel can be formulated as a nonparametric Bayesian Gaussian mixtures model. Based on this, we propose a double-infinite Gaussian mixture model in RFF by placing the Dirichlet process prior. It takes full advantage of high flexibility on the number of components and has the capability of approximating indefinite kernels on a wide scale. In model inference, we develop a non-conjugate variational algorithm with a sub-sampling scheme for posterior inference. It allows for the non-conjugate case in our model and is quite efficient due to the sub-sampling strategy. Experimental results on several large classification datasets demonstrate the effectiveness of our nonparametric Bayesian model for indefinite kernel approximation when compared to other representative random features based methods.**

***Index Terms*—random Fourier features, indefinite kernel, variational inference, kernel approximation**

I. Introduction

Kernel methods [1], [2], [3] have enjoyed tremendous success in statistical machine learning with numerous applications such as classification [4], regression [5], and dimensionality reduction [6]. Whilst a distinct bottleneck of kernel methods is their limited scalability in large datasets, i.e., the huge storage and significant computational cost of the kernel matrix. Given N observations, storing the kernel matrix often needs $\mathcal{O}(N^2)$ space, and takes about $\mathcal{O}(N^2 d)$ operations, where d is the dimension. To make kernel methods scalable, kernel approximation is a powerful technique by mapping input features into a new space. And accordingly, an efficient linear learner can be well trained in the transformed space while retaining the expressive power of nonlinear methods.

To overcome poor scaling in N, several routes have been explored. On the one hand, a straightforward way is employing the divide and conquer approach [7], [8]. It decomposes the full problem into several smaller easy-to-solve subproblems to accelerate the solving process. On the other hand, random projections are widely-applicable and commonly used tactics to seek for a low-rank approximation, either data-dependent or data-independent. The data-dependent approaches approximate the kernel matrix by greedy basis selection techniques [9], incomplete Cholesky decomposition [10], or Nyström methods [11]. In data-independent techniques, the kernel function is directly approximated by an explicit map, which is sampled from a distribution independent of training data. Most approaches that follow this idea are based on random Fourier features (RFF)[1] [14], and have attracted significant attention to scale up kernel methods, such as SVM [15], Gaussian process regression [16], kernel PCA [17], and randomized CCA [18].

The theoretical foundation behind RFF is demonstrated by Bochner's theorem [19], i.e., any bounded, continuous, shift-invariant, and positive definite (PD) function can be expressed as the Fourier transform of a non-negative measure $\rho(\boldsymbol{w})$. However, Bochner's theorem requires the kernel to exhibit two properties: 1) shift-invariance, i.e. $\mathcal{K}(\boldsymbol{x},\boldsymbol{y}) = \mathcal{K}(\boldsymbol{x}-\boldsymbol{y})$ and 2) positive definiteness. These two conditions exclude many widely-used kernels such as dot-product kernels and indefinite kernels (real, symmetric, but not PD) [20], [21], [22]. For instance, the polynomial kernel and the Hellinger's kernel [23] are two popular dot-product kernels that do not satisfy the shift-invariant condition. Indefinite kernels include the hyperbolic tangent kernel [24], the TL1 kernel [25], and Gaussian kernels with a geodesic distance on the manifold [26]. Moreover, dot-product kernels are commonly used on ℓ_2-normalized data to avoid the unboundedness [27], [28], so they can be reformulated as shift-invariant but not always PD on the unit sphere, i.e., $\langle \boldsymbol{x},\boldsymbol{y}\rangle = 1 - 0.5\|\boldsymbol{x}-\boldsymbol{y}\|_2^2$.

Jeffery Pennington et al. [29] theoretically demonstrate that the Fourier transform of a polynomial kernel on the unit

This work was supported in part by the National Natural Science Foundation of China under Grant 61572315, Grant 61876107, Grant 61603248, Grant 11571078, Grant 11631015, Grant U1803261, in part by 973 Plan, China (2015CB856004), in part by Program of Shanghai Subject Chief Scientist (18XD1400700), in part by ERC Advanced Grant E-DUALITY (787960) and KU Leuven CoE PFV/10/002, FWO G0A4917N. (*Corresponding authors: Jie Yang and Xiaolin Huang.*)

F. Liu is with Institute of Image Processing and Pattern Recognition, Shanghai Jiao Tong University, Shanghai 200240, P.R. China, and also with the Department of Electrical Engineering (ESAT-STADIUS), KU Leuven, B-3001 Leuven, Belgium (e-mail: lfhsgre@outlook.com).

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

L. Shi is with Shanghai Key Laboratory for Contemporary Applied Mathematics and School of Mathematical Sciences, Fudan University, Shanghai, 200433, P.R. China (email: leishi@fudan.edu.cn).

J.A.K. Suykens is with the Department of Electrical Engineering (ESAT-STADIUS), KU Leuven, B-3001 Leuven, Belgium (email: johan.suykens@esat.kuleuven.be).

[1] Some recent works [12], [13] are data-dependent.

sphere is not a non-negative function, which is the obstruction to RFF. They empirically use ten Gaussians to approximate the polynomial kernel with spherical random features, and employ a grid-search scheme for parameter tuning. This way is similar to a Gaussian mixture model (GMM) [30] for density estimation, but we need to carefully consider the following two issues. First, the number of Gaussian components is usually ad-hoc pre-defined or manually specified. This is a real limitation as it significantly effects the approximation performance of indefinite kernels. Actually, it is difficult to argue that the number of Gaussian mixtures eventually runs up against some finite bound and remains fixed. We expect to infer the number of Gaussian components needed from data instead of ungrounded guesses. Second, there are numerous parameters in GMM to be estimated, including the mixture coefficients, the mean vector, the covariance of each Gaussian model, and the number of Gaussian components. An efficient parameter estimation technique without heuristic pruning should be developed for model inference, especially when more Gaussians are taken into consideration.

In this paper, we propose a fully non-parametric Bayesian model for approximating non-Bochner kernels (including the dot-product kernel on the unit sphere and shift-invariant indefinite kernel). In our framework, by the Kolmogorov decomposition scheme, an indefinite kernel in the Reproducing Kernel Kreĭn Spaces (RKKS) [20], [31] can be decomposed into the difference of two unknown PD kernels. The spectral distribution of each underlying PD kernel is modeled by an infinite Gaussian mixture model, resulting in a Double-Infinite Gaussian Mixture Model in RFF, termed as RFF-DIGMM. To be specific, our model treats the random frequency $\boldsymbol{w}$ as a latent parameter for each underlying PD kernel, and places a Dirichlet process (DP) prior on it. This makes our random features based framework flexible to indefinite kernel approximation. In model inference, we develop a non-conjugate variational inference method to infer the posterior distribution due to the non-conjugate random frequency $\boldsymbol{w}$ in RFF-DIGMM model. Further, a sub-sampling scheme is used to accelerate the inference process.

Formally, the contributions are summarized as follows.

1) In light of the Kolmogorov decomposition scheme, we propose a Double-Infinite Gaussian Mixture Model for shift-invariant indefinite kernel approximation via random features. As a non-parametric Bayesian model, our model takes full advantage of high flexibility on the number of components and has capability of approximating indefinite kernels in RKKS on a wide scale.
2) In the proposed RFF-DIGMM model, we design a non-conjugate variational inference algorithm with a sub-sampling scheme to infer the non-conjugate posterior distribution. The developed inference algorithm is feasible and efficient to accelerate the inference process for our non-conjugate model.
3) Experimental results illustrate that our RFF-DIGMM model is flexible to approximate indefinite kernels on a wide scale. Furthermore, its application to classification tasks on several large datasets demonstrates the superiority of our RFF-DIGMM model when compared to other representative random features based algorithms.

The remainder of the paper is organized as follows. Section II briefly introduces preliminaries of random Fourier features and the stick-breaking construction for Dirichlet process. Section III presents the proposed RFF-DIGMM model. The non-conjugate variational inference algorithm is given in Section IV. Section V shows the evaluation results of the proposed RFF-DIGMM model with other representative methods on several popular benchmarks. Finally, the conclusion is drawn in Section VI.

II. Preliminaries

This section briefly introduces the rationale of random Fourier features [14], [32] and stick-breaking construction for Dirichlet process [33], [34]. Reviewing these two approaches will help to understand our double-infinite Gaussian mixtures model in RFF. Let $\mathcal{D} = \{\boldsymbol{x}_n\}_{n=1}^N$ be the sample set with N training examples with $\boldsymbol{x}_n \in \mathcal{X} \subseteq \mathbb{R}^d$. Let $\mathcal{K}(\cdot,\cdot)$ be a positive definite kernel function endowed in the Reproducing Kernel Hilbert Space $\mathcal{H}$, and $\boldsymbol{K} = [\mathcal{K}(\boldsymbol{x}_i, \boldsymbol{x}_j)]_{N\times N}$ be the kernel matrix sampled from $\mathcal{D}$. The theoretical foundation of RFF relies on Bochner's celebrated characterization of positive definite functions.

Theorem 1. *(Bochner's theorem [19]) A continuous and shift-invariant function $\mathcal{K} : \mathbb{R}^d \to \mathbb{R}$ is positive definite if and only if it is the Fourier transform of a finite nonnegative Borel measure $\rho(\boldsymbol{w})$ on $\mathbb{R}^d$.*

A consequence of Bochner's theorem is that any shift-invariant and PD kernel can be interpreted by

$$\begin{aligned}\mathcal{K}(\boldsymbol{x}-\boldsymbol{y}) &= \int_{\mathbb{R}^d} p(\boldsymbol{w}) \exp\big(\mathrm{i}\boldsymbol{w}^\top(\boldsymbol{x}-\boldsymbol{y})\big)\mathrm{d}\boldsymbol{w} \\ &= \mathbb{E}_{\boldsymbol{w}\sim\rho(\boldsymbol{w})}\big[\exp(\mathrm{i}\boldsymbol{w}^\top\boldsymbol{x})\exp(\mathrm{i}\boldsymbol{w}^\top\boldsymbol{y})^*\big]\,,\end{aligned} \tag{1}$$

where the symbol $\boldsymbol{x}^*$ denotes the complex conjugate of $\boldsymbol{x}$ and $\rho(\boldsymbol{w})$ can be scaled to a normalized density by setting $\mathcal{K}(0) = 1$. By Monte Carlo integration, the kernel $\mathcal{K}$ can be approximated by

$$\mathcal{K}(\boldsymbol{x}-\boldsymbol{y}) \approx \frac{1}{M}\sum_{m=1}^{M} \exp(\mathrm{i}\boldsymbol{w}_m^\top\boldsymbol{x})\exp(\mathrm{i}\boldsymbol{w}_m^\top\boldsymbol{y})^*\,, \tag{2}$$

where $\boldsymbol{w}_m$ is sampled i.i.d from $\mathcal{P}$ with the density $\rho(\boldsymbol{w})$. In particular, since the kernel $\mathcal{K}$ is real-valued in most cases, the imaginary part of Eq. (2) can be discarded, i.e.

$$\mathcal{K}(\boldsymbol{x}-\boldsymbol{y}) \approx \varphi^\top(\boldsymbol{x})\varphi(\boldsymbol{y}), \text{with } \varphi(\boldsymbol{x}) \triangleq \\ \frac{1}{\sqrt{M}}\big[\cos(\boldsymbol{w}_1^\top\boldsymbol{x}),\cdots,\cos(\boldsymbol{w}_M^\top\boldsymbol{x}), \sin(\boldsymbol{w}_1^\top\boldsymbol{x}),\cdots,\sin(\boldsymbol{w}_M^\top\boldsymbol{x})\big]^\top, \tag{3}$$

where $\varphi(\cdot) : \mathbb{R}^d \mapsto \mathbb{R}^M$ is the random feature mapping, and $\varphi^\top(\boldsymbol{x})\varphi(\boldsymbol{y})$ is the unbiased estimation of $\mathcal{K}(\boldsymbol{x},\boldsymbol{y})$. Hence, by random features, the storage and computational complexity can be reduced to $\mathcal{O}(NM)$ and $\mathcal{O}(NMd)$, respectively. Recent works on random features aim to improve the approximation quality by Quasi-Monte Carlo sampling [35], random orthogonal matrix [32], or decrease the time and space complexity by Fastfood [36], quadrature-based features [37]. However, these algorithms mainly focus on shift-invariant and PD kernels, and cannot be directly applied to non-Bochner kernels. Only a few literature based on random features are able to deal with a polynomial kernel by the Maclaurin's approximation [38] and tensor sketching [27], or indefinite kernel approximation by finite Gaussian mixtures [29].

Next, we briefly review the stick-breaking construction for Dirichlet process (DP) [39]. DP is a stochastic process over discrete probability measures, i.e., atoms, with countably infinite support. It is widely used in Bayesian nonparametric models of data, particularly in Dirichlet process mixture models [40]. Mathematically, let G be a distribution over the probability space Θ, α be a positive real scalar, and H be a base measure over Θ. If any r partitions $(A_1, A_2, \cdots, A_r)$ of the corresponding probability space obey a Dirichlet distribution, then the distribution $(G(A_1), G(A_2), \cdots, G(A_r))$ is a Dirichlet process

$$(G(A_1),G(A_2),\cdots,G(A_r)) \sim \mathrm{Dir}(\alpha H(A_1),\alpha H(A_2),\cdots,\alpha H(A_r)),$$

where r is a natural number [34] and α is the concentration parameter. We denote it as $G \sim \mathcal{DP}(\alpha, H)$.

To build a DP, one representative strategy is stick-breaking construction [41]. Given a unit-length stick $(0,1)$, we first draw $\beta_1 \sim \mathrm{Beta}(1,\alpha_0)$, set $\theta_1 \triangleq \beta_1$, and pick the fraction $1-\beta_1$ as the remainder of the stick. And then, we draw $\beta_2 \sim \mathrm{Beta}(1,\alpha_0)$, and assign $\theta_2 \triangleq \beta_2(1-\beta_1)$. Repeating this procedure, we have Dirichlet process mixtures with stick-braking representation, i.e., the random measure G is associated with a Dirichlet process $\mathcal{DP}(G_0, \alpha_0)$ with respect to base distribution G_0 and concentration parameter α_0. Mathematically, we have

$$G = \sum_{k=1}^{\infty} \theta_k(\boldsymbol{\beta})\delta_{\Phi_k}, \quad \theta_k(\boldsymbol{\beta}) = \beta_k \prod_{s=1}^{k-1}(1-\beta_s)\,, \tag{4}$$

with $\Phi_k \sim G_0$ and $\beta_k|\alpha_0 \sim \mathrm{Beta}(1,\alpha_0)$. The notation δ_{Φ_k} is the Kronecker delta function, of which the value is 1 at location Φ_k and 0 elsewhere. It can be found that, G is discrete almost surely, i.e., the support of G consists of a countably infinite set of atoms, which are drawn independently from G_0. Since the distributions sampled from a DP are discrete almost surely, data generated from a DP mixture can be partitioned into different groups according to the distinct values of the sampled distributions. As a result, the whole model serves as a mixture model, in which the number of components is random and grows as new data are observed. For more details on the nonparametric Bayesian model and its construction, we refer the reader to [34] and [42].

III. Model Description

In this section, we present the formulation of our RFF-DIGMM model and its graphical model representation.

A. Kolmogorov decomposition for indefinite kernels

In theory, a functional space spanned by indefinite kernels does not belong to the Reproducing Kernel Hilbert Spaces (RKHS) [1], [43]. To investigate indefinite kernels, we need Kreĭn spaces defined as follows.

Definition 1. *(Kreĭn space [31]) An inner product space is a Kreĭn space $\mathcal{H}_{\mathcal{K}}$ if there exist two Hilbert spaces $\mathcal{H}^+$ and $\mathcal{H}^-$ such that*

- *All $f \in \mathcal{H}_{\mathcal{K}}$ can be decomposed into $f = f^+ + f^-$, where $f^+ \in \mathcal{H}^+$ and $f^- \in \mathcal{H}^-$, respectively.*
- *$\forall f, g \in \mathcal{H}_{\mathcal{K}}$, $\langle f, g\rangle_{\mathcal{H}_{\mathcal{K}}} = \langle f^+, g^+\rangle_{\mathcal{H}_+} - \langle f^-, g^-\rangle_{\mathcal{H}_-}$.*

If $\mathcal{H}_+$ and $\mathcal{H}_-$ are two RKHSs, the Kreĭn space $\mathcal{H}_{\mathcal{K}}$ is a RKKS associated with a unique indefinite reproducing kernel $\mathcal{K}$ such that the reproducing property holds, i.e., $\forall f \in \mathcal{H}_{\mathcal{K}},\ f(\boldsymbol{x}) = \langle f, k(\boldsymbol{x},\cdot)\rangle_{\mathcal{H}_{\mathcal{K}}}$. To link indefinite kernels of RKKS to RKHS, we present a useful proposition as follows.

Proposition 1. *(Proposition 2.1 in [44]) An indefinite reproducing kernel $\mathcal{K}$ associated with a RKKS admits a Kolmogorov decomposition*

$$\mathcal{K} = \mathcal{K}^+ - \mathcal{K}^-\,,$$

with two positive definite kernels $\mathcal{K}^+$ and $\mathcal{K}^-$.

Typical examples of indefinite kernels that admit Kolmogorov decomposition include a wide range of commonly used indefinite kernels, such as a linear combination of PD kernels, and conditionally PD kernels. Hence, approximating an indefinite kernel $\mathcal{K}$ in RKKS by random features can be formulated as conducting random feature mappings for two underlying PD kernels $\mathcal{K}^+$ and $\mathcal{K}^-$.

Although the above proposition presents the existence of a Kolmogorov decomposition for an indefinite kernel in RKKS, it does not provide a specific decomposition result for $\mathcal{K}^+$ and $\mathcal{K}^-$. In this case, what we only have is the indefinite kernel $\mathcal{K}$ and its associated indefinite kernel matrix $\boldsymbol{K}$ on the sample set $\mathcal{D}$. An intuitive way is to conduct an eigenvalue decomposition for $\boldsymbol{K}$, i.e., $\boldsymbol{K} = \boldsymbol{U}^\top \boldsymbol{\Gamma} \boldsymbol{U}$, where $\boldsymbol{U}$ is an orthogonal matrix and the diagonal matrix is $\boldsymbol{\Gamma} = \mathrm{diag}(\lambda_1, \lambda_2, \cdots, \lambda_N)$ with eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq 0 \geq \cdots \geq \lambda_N$. Without loss of generality, we assume that the first s eigenvalues are nonnegative and the remaining $N-s$ ones are negative. Hence, $\boldsymbol{K}$ can be decomposed as $\boldsymbol{K} = \boldsymbol{K}^+ - \boldsymbol{K}^-$ with the following formulation

$$\begin{cases} \boldsymbol{K}^+ = \boldsymbol{U}^\top \mathrm{diag}(\lambda_1+\tau, \ldots, \lambda_s+\tau)\boldsymbol{U} \\ \boldsymbol{K}^- = \boldsymbol{U}^\top \mathrm{diag}(\tau-\mu_{N-s+1}, \ldots, \tau-\mu_N)\boldsymbol{U}\,, \end{cases} \tag{5}$$

where τ is to ensure that these two matrices $\boldsymbol{K}^+$ and $\boldsymbol{K}^-$ are positive definite. Further, to speed up the computational efficiency in large scale situations, we only consider a subset of training examples to conduct eigenvalue decomposition. That is, given two sub-matrices from $\boldsymbol{K}^+$ and $\boldsymbol{K}^-$, our target is to obtain random feature mappings for $\mathcal{K}^+$ and $\mathcal{K}^-$ by the proposed RFF-DIGMM model.

B. Graphical Model Representation for RFF-DIGMM

Bochner's theorem shows that the characteristic function (i.e., the inverse Fourier transformation) of a continuous distribution $\mathcal{P}$ with its pdf $\rho(\boldsymbol{w})$ is associated with a shift-invariant and PD kernel [45]. For example, suppose that $\rho(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is a Gaussian distribution with the mean vector $\boldsymbol{\mu}$ and the covariance matrix $\boldsymbol{\Sigma}$, its characteristic function is a shift-invariant kernel $\mathcal{K}(\Delta) = \exp(\mathrm{i}\boldsymbol{\mu}^\top\Delta - \frac{1}{2}\Delta^\top\boldsymbol{\Sigma}\Delta)$ with

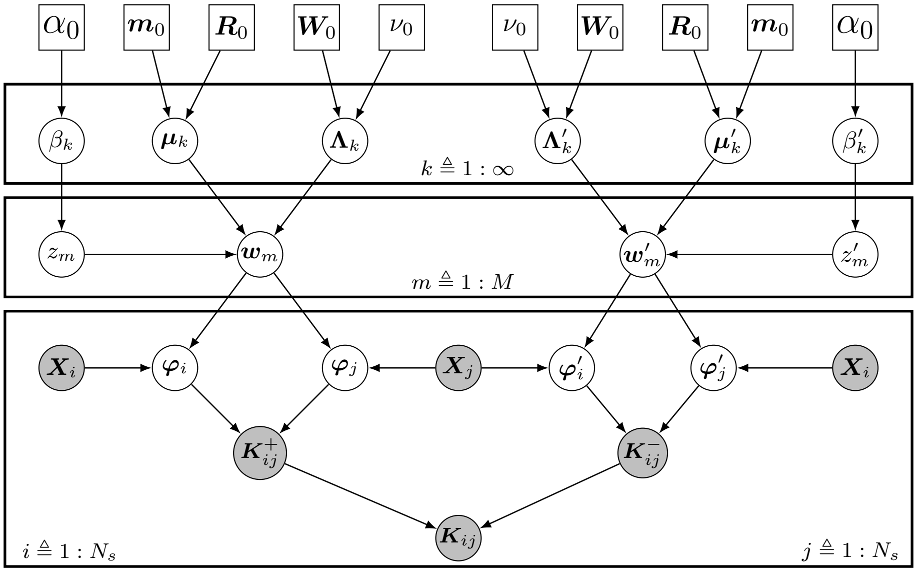


Fig. 1: Graphical model representation of RFF-DIGMM.

$\Delta := \boldsymbol{x} - \boldsymbol{y}$. That is to say, a Gaussian distribution and its characteristic function define a Gaussian kernel. Considering that GMM is a universal approximator for any continuous distribution [46] in density estimation, the spectral distribution $\mathcal{P}$ can be well approximated by GMM. From the kernel learning perspective, this mixture modeling is able to yield a general PD kernel, which provides a justification to obtain random feature mappings for $\mathcal{K}^+$ and $\mathcal{K}^-$, respectively.

For the underlying PD kernel $\mathcal{K}^+$, since the number of its corresponding nonnegative Borel measure $\rho(\boldsymbol{w})$ is not a prior known, we posit it as infinite, namely

$$\rho(\boldsymbol{w}) = \sum_{k=1}^{\infty} \theta_k \mathcal{N}\big(\boldsymbol{w}|\boldsymbol{\mu}_k, \boldsymbol{\Lambda}_k^{-1}\big)\,, \tag{6}$$

where $\boldsymbol{\mu}_k$ and $\boldsymbol{\Lambda}_k$ are the mean vector and precision matrix of each Gaussian, respectively. According to Plancherel's theorem [47], the expression of $\rho(\boldsymbol{w})$ with infinite components in Eq. (6) is able to approximate any shift invariant PD kernel. It relates spectral accuracies to the original domain by the following characteristic function

$$\mathcal{K}^+(\boldsymbol{x}-\boldsymbol{y}) = \sum_{k=1}^{\infty} \theta_k \exp\Big(\mathrm{i}\boldsymbol{\mu}_k^\top(\boldsymbol{x}-\boldsymbol{y}) - \frac{1}{2}(\boldsymbol{x}-\boldsymbol{y})^\top \boldsymbol{\Lambda}_k(\boldsymbol{x}-\boldsymbol{y})\Big).$$

In practical use, the kernel is often real-valued, so we consider the real part of the above equation

$$\mathcal{K}^+(\Delta) = \sum_{k=1}^{\infty} \theta_k \exp\Big(-\frac{1}{2}\Delta^\top \boldsymbol{\Lambda}_k \Delta\Big) \cos\big(\boldsymbol{\mu}_k^\top \Delta\big)\,,$$

with $\Delta := \boldsymbol{x} - \boldsymbol{y}$. In this case, the PD kernel $\mathcal{K}^+$ can be approximated by $\mathcal{K}^+(\boldsymbol{x}, \boldsymbol{y}) \approx \varphi^\top(\boldsymbol{x})\varphi(\boldsymbol{y})$, where $\varphi(\boldsymbol{x})$ is as in Eq. (3).

Likewise, for $\mathcal{K}^-$, its corresponding nonnegative Borel measure $\rho'(\boldsymbol{w}')$ is formulated as

$$\rho'(\boldsymbol{w}') = \sum_{k=1}^{\infty} \theta_k \mathcal{N}\big(\boldsymbol{w}'|\boldsymbol{\mu}'_k, \boldsymbol{\Lambda}'^{-1}_k\big)\,, \tag{7}$$

and its characteristic function is

$$\mathcal{K}^-(\boldsymbol{x}-\boldsymbol{y}) = \sum_{k=1}^{\infty} \theta'_k \exp\Big(\mathrm{i}\boldsymbol{\mu}'^\top_k(\boldsymbol{x}-\boldsymbol{y}) - \frac{1}{2}(\boldsymbol{x}-\boldsymbol{y})^\top \boldsymbol{\Lambda}'_k(\boldsymbol{x}-\boldsymbol{y})\Big)\,.$$

In this case, the PD kernel $\mathcal{K}^-$ can be approximated by

$$\mathcal{K}^-(\boldsymbol{x}-\boldsymbol{y}) \approx \varphi'^\top(\boldsymbol{x})\varphi'(\boldsymbol{y}), \text{with } \varphi'(\boldsymbol{x}) \triangleq \frac{1}{\sqrt{M}}\big[\cos(\boldsymbol{w}'^\top_1\boldsymbol{x}),\cdots,\cos(\boldsymbol{w}'^\top_M\boldsymbol{x}), \sin(\boldsymbol{w}'^\top_1\boldsymbol{x}),\cdots,\sin(\boldsymbol{w}'^\top_M\boldsymbol{x})\big]^\top\,. \tag{8}$$

Therefore, the expression of $\rho(\boldsymbol{w})$ in Eq. (6) and $\rho'(\boldsymbol{w}')$ in Eq. (7) with infinite components provide adequate flexibility to find a good approximation of $\mathcal{K}$ from a broad class.

The graphical model representation of our RFF-DIGMM model is shown in Fig. 1. In our model, to speed up the computational efficiency and to reduce the memory storage, we randomly select N_s examples from the training set $\mathcal{D}$, resulting in the sketch $\mathcal{D}_s$. Similar to [48], [49], our model works between sub-sampling the training set and adjusting the hidden structure for parameter estimation based on the sketch $\mathcal{D}_s$. Thereby, finding a good approximation to a non-Bochner kernel over N_s observations can be represented as

$$K_{ij} = K^+_{ij} - K^-_{ij} = \varphi^\top(\boldsymbol{x}_i)\varphi(\boldsymbol{x}_j) - \varphi'^\top(\boldsymbol{x}_i)\varphi'(\boldsymbol{x}_j) + \epsilon, \quad \forall \boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{D}_s \text{ and } i \neq j\,, \tag{9}$$

with $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$. The two random feature mappings φ and φ' satisfy $\mathcal{K}^+(\boldsymbol{x}_i - \boldsymbol{x}_j) \approx \varphi^\top(\boldsymbol{x}_i)\varphi(\boldsymbol{x}_j)$ and $\mathcal{K}^-(\boldsymbol{x}_i - \boldsymbol{x}_j) \approx \varphi'^\top(\boldsymbol{x}_i)\varphi'(\boldsymbol{x}_j)$, respectively. It is important to point out that, on each trial, we randomly sample two examples $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ ($i \neq j$) without replacement from $\mathcal{D}_s$ to construct K_{ij}, and directly set $K_{ii} = 1$. By doing so, we are able to avoid the situation when the pair example $(\boldsymbol{x}_i, \boldsymbol{x}_j, K_{ij})$ is not mutually pairwise independent [50].

In our RFF-DIGMM model, since the explicit feature mappings φ and φ' in Eq. (9) are determined by $\rho(\boldsymbol{w})$ and

$\rho'(\boldsymbol{w}')$, respectively, the distributions of K_{ij}^{+} and K_{ij}^{+} are

$$p\big(K_{ij}^{+}|(\boldsymbol{x}_i,\boldsymbol{x}_j),\varphi\big)\sim\mathcal{N}\bigg(\frac{1}{M}\sum_{m=1}^{M}\cos\Big(\boldsymbol{w}_m^{\top}(\boldsymbol{x}_i-\boldsymbol{x}_j)\Big),\sigma_\epsilon^2\bigg)\,,$$

$$p\big(K_{ij}^{-}|(\boldsymbol{x}_i,\boldsymbol{x}_j),\varphi'\big)\sim\mathcal{N}\bigg(\frac{1}{M}\sum_{m=1}^{M}\cos\Big(\boldsymbol{w}_m'^{\top}(\boldsymbol{x}_i-\boldsymbol{x}_j)\Big),\sigma_\epsilon^2\bigg)\,.$$

The random frequencies $\boldsymbol{w}_m$ and $\boldsymbol{w}'_m$ over the input space for a mixture component are given by

$$\begin{cases} p(\boldsymbol{w}_m|z_m=k,\boldsymbol{\mu}_k,\boldsymbol{\Lambda}_k)\sim\mathcal{N}(\boldsymbol{w}_m|\boldsymbol{\mu}_k,\boldsymbol{\Lambda}_k^{-1})\,,\\ p(\boldsymbol{w}'_m|z'_m=k,\boldsymbol{\mu}'_k,\boldsymbol{\Lambda}'_k)\sim\mathcal{N}(\boldsymbol{w}'_m|\boldsymbol{\mu}'_k,\boldsymbol{\Lambda}_k'^{-1})\,,\end{cases}$$

where z_m, z'_m are two latent variables that assign the indices of the parameter associated with $\boldsymbol{w}_m$ and $\boldsymbol{w}'_m$. The mean vectors $\boldsymbol{\mu}_k$, $\boldsymbol{\mu}'_k$ and the precision matrices $\boldsymbol{\Lambda}_k$, $\boldsymbol{\Lambda}'_k$ are further specified by Gaussian distribution priors and Normal-Wishart distribution priors with the same hyper-parameters, respectively. To be specific, the corresponding priors are

$$\boldsymbol{\mu}_k,\boldsymbol{\mu}'_k\sim\mathcal{N}\big(\boldsymbol{m}_0,\boldsymbol{R}_0^{-1}\big)\,,\quad\boldsymbol{\Lambda}_k,\boldsymbol{\Lambda}'_k\sim\mathcal{W}(\boldsymbol{W}_0,\nu_0)\,.\qquad(10)$$

Besides, the distribution of z_m can be regarded as a multinomial distribution with parameters $\{\theta_k\}_{k=1}^{\infty}$ by the following formulation

$$p(z_m|\beta_k)=\prod_{k=1}^{\infty}(1-\beta_k)^{1[z_m>k]}\beta_k^{1[z_m=k]}\,,\qquad(11)$$

where β_k is given by Eq. (4), determining the mixing proportions $\{\theta_k\}_{k=1}^{\infty}$. The prior for $\{\theta_k\}_{k=1}^{\infty}$ is a DP prior built by stick-breaking construction, so we define it by the stick-breaking distribution $\boldsymbol{\theta}\sim\mathrm{GEM}(\alpha_0)$, where GEM (Griffiths-Engen-McCloskey) is the stick breaking prior [41]. The mixing proportions $\{\theta_k\}_{k=1}^{\infty}$ can be regarded as a sequence of sticks with lengths, satisfying $\sum_{k=1}^{\infty}\theta_k=1$. The product $\prod_{s=1}^{k-1}(1-\beta_s)$ denotes the previous remaining length of the stick, and multiplication by β_s gives the length of the stick currently broken off. Hence, Eq. (11) can be formulated as

$$z_m|\{\beta_1,\beta_2,\cdots,\beta_\infty\}\sim\mathrm{Mult}\big(\boldsymbol{\beta}\big)\,,$$

where Mult denotes the multinomial distribution. Similarly, z'_m is subject to

$$p(z'_m|\beta'_k)=\prod_{k=1}^{\infty}(1-\beta'_k)^{1[z'_m>k]}\beta_k'^{1[z'_m=k]}\,.$$

Finally, the complete generative process is given below.

1) Draw the mixing proportions $\{\theta_i\}_{i=1}^{\infty}:\boldsymbol{\theta}\sim\mathrm{GEM}(\alpha_0)$ and $\{\theta'_i\}_{i=1}^{\infty}:\boldsymbol{\theta}'\sim\mathrm{GEM}(\alpha_0)$.
2) Draw the mixture components, for $k=1:\infty$
   - draw $\boldsymbol{\mu}_k,\boldsymbol{\mu}'_k\sim\mathcal{N}\big(\boldsymbol{m}_0,\boldsymbol{R}_0^{-1}\big)$.
   - draw $\boldsymbol{\Lambda}_k,\boldsymbol{\Lambda}'_k\sim\mathcal{W}(\boldsymbol{W}_0,\nu_0)$.
3) For each random frequency index $m=1,2,\cdots,M$
   - draw the indicate labels $z_m|\{\beta_1,\beta_2,\cdots,\beta_\infty\}\sim\mathrm{Mult}(\boldsymbol{\theta}(\boldsymbol{\beta}))$ and $z'_m|\{\beta'_1,\beta'_2,\cdots,\beta'_\infty\}\sim\mathrm{Mult}(\boldsymbol{\theta}'(\boldsymbol{\beta}'))$.
   - draw the random feature vectors $\boldsymbol{w}_m\sim\mathcal{N}(\boldsymbol{w}_m|z_m=k,\boldsymbol{\mu}_k,\boldsymbol{\Lambda}_k^{-1})$ and $\boldsymbol{w}'_m\sim\mathcal{N}(\boldsymbol{w}'_m|z'_m=k,\boldsymbol{\mu}'_k,\boldsymbol{\Lambda}_k'^{-1})$.
4) For any two selected training examples $\boldsymbol{x}_i,\boldsymbol{x}_j\in\mathcal{D}_s$
   - compute $\varphi(\boldsymbol{x}_i)$, $\varphi(\boldsymbol{x}_j)$, $\varphi'(\boldsymbol{x}_i)$, and $\varphi'(\boldsymbol{x}_j)$ by Eq. (3).
   - draw an observation $K_{ij}\sim\mathcal{N}\Big(\varphi^{\top}(\boldsymbol{x}_i)\varphi(\boldsymbol{x}_j)-\varphi'^{\top}(\boldsymbol{x}_i)\varphi'(\boldsymbol{x}_j),\sigma_\epsilon^2\Big)$.

After conducting the generative process of our RFF-DIGMM model, we need to infer the associated parameters with respect to $\rho(\boldsymbol{w})$ and $\rho'(\boldsymbol{w}')$. For $\rho(\boldsymbol{w})$, defining parameter sets $\tilde{\boldsymbol{\beta}}=\{\beta_1,\beta_2,\cdots,\beta_\infty\}$, $\tilde{\boldsymbol{\mu}}=\{\boldsymbol{\mu}_1,\boldsymbol{\mu}_2,\cdots,\boldsymbol{\mu}_\infty\}$, $\tilde{\boldsymbol{\Lambda}}=\{\boldsymbol{\Lambda}_1,\boldsymbol{\Lambda}_2,\cdots,\boldsymbol{\Lambda}_\infty\}$, latent variable sets $\tilde{\boldsymbol{w}}=\{\boldsymbol{w}_1,\boldsymbol{w}_2,\cdots,\boldsymbol{w}_M\}$, $\tilde{\boldsymbol{z}}=\{z_1,z_2,\cdots,z_M\}$, the hidden variable set is given by $\Omega=\{\tilde{\boldsymbol{\beta}},\tilde{\boldsymbol{\mu}},\tilde{\boldsymbol{\Lambda}},\tilde{\boldsymbol{z}},\tilde{\boldsymbol{w}}\}$. As illustrated by the graphical model shown in Fig. 1, the joint distribution of all the random variables with respect to ρ is given by

$$p(\mathcal{D}_s,\Omega)=p(\tilde{\boldsymbol{\beta}})p(\tilde{\boldsymbol{\mu}})p(\tilde{\boldsymbol{\Lambda}})\prod_{m=1}^{M}p(z_m|\tilde{\boldsymbol{\beta}})p(\boldsymbol{w}_m|z_m,\tilde{\boldsymbol{\mu}},\tilde{\boldsymbol{\Lambda}})$$
$$\times\prod_{i,j=1,i\neq j}^{N_s}p(K_{ij}^{+}|(\boldsymbol{x}_i,\boldsymbol{x}_j),\tilde{\boldsymbol{w}})\,,$$

where the notations are $p(\tilde{\boldsymbol{\beta}})=\prod_{k=1}^{\infty}p(\beta_k)$, $p(\tilde{\boldsymbol{\mu}})=\prod_{k=1}^{\infty}p(\boldsymbol{\mu}_k)$, $p(\tilde{\boldsymbol{\Lambda}})=\prod_{k=1}^{\infty}p(\boldsymbol{\Lambda}_k)$, $p(\tilde{\boldsymbol{z}})=\prod_{m=1}^{M}p(z_m)$, and $p(\tilde{\boldsymbol{w}})=\prod_{m=1}^{M}p(\boldsymbol{w}_m)$. Accordingly, we have

$$p(\mathcal{D}_s,\Omega)=\prod_{k=1}^{\infty}p(\beta_k|\alpha_0)p(\boldsymbol{\mu}_k|\boldsymbol{m}_0,\boldsymbol{R}_0)p(\boldsymbol{\Lambda}_k|\boldsymbol{W}_0,\nu_0)$$
$$\times\prod_{m=1}^{M}p(z_m|\tilde{\boldsymbol{\beta}})p(\boldsymbol{w}_m|z_m,\tilde{\boldsymbol{\mu}},\tilde{\boldsymbol{\Lambda}})\prod_{i,j=1,i\neq j}^{N_s}p(K_{ij}^{+}|(\boldsymbol{x}_i,\boldsymbol{x}_j),\tilde{\boldsymbol{w}})\,.$$

Likewise, the joint distribution of all the random variables with respect to ρ' is given by

$$p(\mathcal{D}_s,\Omega')=\prod_{k=1}^{\infty}p(\beta'_k|\alpha_0)p(\boldsymbol{\mu}'_k|\boldsymbol{m}_0,\boldsymbol{R}_0)p(\boldsymbol{\Lambda}'_k|\boldsymbol{W}_0,\nu_0)$$
$$\times\prod_{m=1}^{M}p(z'_m|\tilde{\boldsymbol{\beta}}')p(\boldsymbol{w}'_m|z'_m,\tilde{\boldsymbol{\mu}}',\tilde{\boldsymbol{\Lambda}}')\prod_{i,j=1,i\neq j}^{N_s}p(K_{ij}^{-}|(\boldsymbol{x}_i,\boldsymbol{x}_j),\tilde{\boldsymbol{w}}')\,,$$

where the variable notations $\Omega'=\{\tilde{\boldsymbol{\beta}}',\tilde{\boldsymbol{\mu}}',\tilde{\boldsymbol{\Lambda}}',\tilde{\boldsymbol{z}}',\tilde{\boldsymbol{w}}'\}$ for ρ' share the similar formulation with the corresponding definitions for ρ. Since the posteriors $p(\Omega|\mathcal{D}_s)$ and $p(\Omega'|\mathcal{D}_s)$ are often intractable, in the next section, we will approximate them using mean-field variational inference.

IV. Inference

In this section, we develop a variant of the mean-field variational inference algorithm to tackle the non-conjugate variable $\boldsymbol{w}$ in our model. Here we take $\rho(\boldsymbol{w})$ as an example to illustrate the inference process. The inference for $\rho'(\boldsymbol{w}')$ can be obtained in a similar way.

A. Truncated DP in Mean-field Approach

Variational inference [33], [51] aims to find a distribution in a simple family that is close to the true posterior distribution $p(\Omega|\mathcal{D}_s)$ by a proxy $q(\Omega)$ with the following decomposition

$$\ln p(\mathcal{D}_s)=\mathcal{L}(q)+\mathrm{KL}(q||p)\,,\qquad(12)$$

where the Kullback-Leibler (KL) divergence is defined as $\mathrm{KL}(q||p) = \int q(\Omega) \ln\{q(\Omega)/p(\Omega|\mathcal{D}_s)\}\mathrm{d}\Omega$, and $\mathcal{L}(q)$ is the lower bound of $\ln p(\mathcal{D}_s)$ with the expression $\mathcal{L}(q) = \int q(\Omega) \ln\{p(\mathcal{D}_s, \Omega)/q(\Omega)\}\mathrm{d}\Omega$. Variational inference can be formulated as minimizing the KL divergence from the variational distribution to the posterior distribution, which is equivalent to maximize the *evidence lower bound* (ELBO).

To formulate the variational posterior $q(\Omega)$, the posterior Dirichlet process is approximated by a truncated stick-breaking representation [52]. That is, given a value T, we set $q(\beta_T = 1) = 1$ to guarantee that the mixture proportions θ_k are zero for $k > T$. Note that the variational distribution is truncated but our model is a full DP and is not truncated. Based on the truncated DP, we adopt the mean-field approximation by the fully factorized variational distribution to approximate $p(\Omega|\mathcal{D}_s)$

$$q(\Omega|\mathcal{D}_s) = \prod_{t=1}^{T-1} q(\beta_t) \prod_{k=1}^{T} q(\boldsymbol{\mu}_k) q(\boldsymbol{\Lambda}_k) \prod_{m=1}^{M} q(\boldsymbol{w}_m) q(z_m)\,.$$

Using the above full factorization formulation, we can solve $q(\Omega|\mathcal{D}_s)$ by maximizing the lower bound $\mathcal{L}(q)$ in Eq. (12). The logarithm of the optimized factor $q^*(\boldsymbol{\vartheta})$ with $\boldsymbol{\vartheta} \in \Omega$ is

$$\ln q^*(\boldsymbol{\vartheta}) = \mathbb{E}_{\Omega\setminus\boldsymbol{\vartheta}} \ln p(\mathcal{D}_s, \Omega) + \text{const}\,, \tag{13}$$

where $\mathbb{E}_{\Omega\setminus\boldsymbol{\vartheta}}$ is the expectation with respect to all other latent variables, and "const" (short for c) denotes a constant that is independent of $\boldsymbol{\vartheta}$. Therefore, using the ELBO and the mean-field family, the posterior approximate is cast as an optimization problem. It can be efficiently solved by a coordinate ascent variational inference [53] and we detail this as follows.

B. Update Variational Factors

The optimization for each variational factor is conducted by the coordinate ascent variational inference. It iteratively optimizes each factor of the mean-field variational density, while holding the others fixed, which climbs the ELBO to a local optimum. Here we just state the results and the derivations can be found in Appendix A.

1) $q(\beta_t)$: We absorb terms in Eq. (13) that are independent of β_t into the additive normalization constant, and then get a Beta posterior approximating distribution

$$\beta_t \sim \mathrm{Beta}\left(1 + \sum_{m=1}^{M} q(z_m = t), \alpha_0 + \sum_{m=1}^{M} q(z_m > t)\right).$$

2) $q(z_m)$: Likewise, we do not consider irrelevant terms of z_m in Eq. (13). Defining $\Xi \triangleq \mathbb{E}_{\boldsymbol{w}_m,\boldsymbol{\mu}_k,\boldsymbol{\Lambda}_k}\big[(\boldsymbol{w}_m - \boldsymbol{\mu}_k)^\top \boldsymbol{\Lambda}_k (\boldsymbol{w}_m - \boldsymbol{\mu}_k)\big]$, $\ln \hbar_{mk} \triangleq \mathbb{E}(\ln \beta_k) + \sum_{t=1}^{k-1} \mathbb{E}[\ln(1 - \beta_t)] + \frac{1}{2}\Big(\mathbb{E} \ln |\boldsymbol{\Lambda}_k| - d \ln(2\pi) - \Xi\Big)$, and scaling $\tilde{\hbar}_{mk} = \frac{\hbar_{mk}}{\sum_{t=1}^{T} \hbar_{mt}}$, we have $q(z_m = k) = \tilde{\hbar}_{mk}$. It means that z_m is chosen according to a multinomial probability distribution.

3) $q(\boldsymbol{\mu}_k)$: Keeping only terms that have functional dependence on $\boldsymbol{\mu}_k$, we get a Gaussian posterior approximating distribution $\boldsymbol{\mu}_k \sim \mathcal{N}\big(\boldsymbol{\mu}_k|\boldsymbol{m}_k, \boldsymbol{R}_k^{-1}\big)$ with the following mean vector and precision matrix

$$\begin{cases} \boldsymbol{m}_k = \boldsymbol{R}_k^{-1}\Big(\boldsymbol{R}_0 \boldsymbol{m}_0 + \mathbb{E}(\boldsymbol{\Lambda}_k) \sum_{m=1}^{M} q(z_m = k) \mathbb{E}(\boldsymbol{w}_m)\Big) \\ \boldsymbol{R}_k = \boldsymbol{R}_0 + \mathbb{E}(\boldsymbol{\Lambda}_k) \sum_{m=1}^{M} q(z_m = k)\,. \end{cases}$$

4) $q(\boldsymbol{\Lambda}_k)$: We only retain some terms with respect to $\boldsymbol{\Lambda}_k$ in Eq. (13), the approximating distribution is $\boldsymbol{\Lambda}_k \sim \mathcal{W}(\boldsymbol{\Lambda}_k|\boldsymbol{W}_k, \nu_k)$ with $\nu_k = \nu_0 + \sum_{m=1}^{M} q(z_m = k)$ and $\boldsymbol{W}_k^{-1}$ is formulated by

$$\boldsymbol{W}_k^{-1} = \boldsymbol{W}_0^{-1} + \sum_{m=1}^{M} q(z_m = k) \mathbb{E}(\boldsymbol{w}_m - \boldsymbol{\mu}_k)(\boldsymbol{w}_m - \boldsymbol{\mu}_k)^\top.$$

5) $q(\boldsymbol{w}_m)$: The equation for solving $\boldsymbol{w}_m$ is a little complex, because $\boldsymbol{w}_m$ is involved in multiple variational factors. Due to the fact that $\boldsymbol{w}_m$ is a non-conjugate variable, here we use the second order Taylor expansion for cosine function, i.e., $\cos\big[\boldsymbol{w}_m^\top(\boldsymbol{x}_i - \boldsymbol{x}_j)\big] \approx 1 - \frac{1}{2}\boldsymbol{w}_m^\top(\boldsymbol{x}_i - \boldsymbol{x}_j)(\boldsymbol{x}_i - \boldsymbol{x}_j)^\top \boldsymbol{w}_m$. Accordingly, we inspect Eq. (13) and read off those terms which involve $\boldsymbol{w}_m$. Defining

$$\boldsymbol{S} \triangleq \sum_{k=1}^{T} [q(z_m = k)\mathbb{E}(\boldsymbol{\Lambda}_k)] + \frac{1}{2\sigma_\epsilon^2} \sum_{i,j=1, i\neq j}^{N_s} (1 - K_{ij}^{+})(\boldsymbol{x}_i - \boldsymbol{x}_j)(\boldsymbol{x}_i - \boldsymbol{x}_j)^\top,$$

we get the posterior approximating distribution for $\boldsymbol{w}_m$

$$\boldsymbol{w}_m \sim \mathcal{N}\left(\boldsymbol{S}^{-1}\left\{\sum_{k=1}^{T} \big[q(z_m = k)\mathbb{E}(\boldsymbol{\Lambda}_k)\mathbb{E}(\boldsymbol{\mu}_k)\big]\right\}, \boldsymbol{S}^{-1}\right).$$

The variational distribution $q(\boldsymbol{w}_m)$ is subject to a Gaussian distribution. Its Gaussian form naturally stems from the Taylor approximation of the cosine function. By Bochner's theorem, we have $\mathbb{E}[\cos(\boldsymbol{w}_m^\top \bar{\boldsymbol{x}})] = \exp(-\|\bar{\boldsymbol{x}}\|^2/2)$ with $\bar{\boldsymbol{x}} := \frac{\boldsymbol{x}_i - \boldsymbol{x}_j}{\sigma}$. Hence, with a proper scale width σ, we can guarantee that $\langle \boldsymbol{w}_m, \boldsymbol{x}_i - \boldsymbol{x}_j \rangle = 0$ with high probability when $\|\bar{\boldsymbol{x}}\|$ approaches to zero, and accordingly the Taylor approximation condition is satisfied. This approximation technique can also be found in Laplace approximation variational inference for non-conjugate models [54]. Unlike Laplace approximation, our variational inference algorithm does not require the exponential family assumption but directly uses the Taylor approximation of the cosine function.

Finally, by repeating the update steps above, we adjust the free variational parameters to approximate the original distribution $p(\Omega|\mathcal{D}_s)$ until convergence. Likewise, the variational approximation for $p(\Omega'|\mathcal{D}_s)$ can be obtained in the similar fashion. The variational inference algorithm for model inference is summarized in Algorithm 1. The convergence results of our model are similar to the Laplace approximation method in [54], which converges to a local optimum of the variational objective. Here we assess convergence by measuring the difference between the two consecutive iterations for $q(\boldsymbol{z})$. This is a common stopping criterion and we set the maximum iteration number IterMAX to 50. We will experimentally verify the convergence of the proposed inference algorithm in Section V-F.

Algorithm 1: Variational inference for RFF-DIGMM.

```
Construct $\mathcal{D}_s$ and the associated sub-kernel matrix $\boldsymbol{K}$.
Obtain $\boldsymbol{K}^+$ and $\boldsymbol{K}^-$ by eigenvalue decomposition for $\boldsymbol{K}$.
Set IterMAX$= 50$, iter $= 0$, initialize variational
  distributions $q(\Omega|\mathcal{D}_s)$ and $q(\Omega'|\mathcal{D}_s)$.
Repeat
    iter = iter + 1;
    for $k = 1$ to $T$ do
        Update $q(\beta_k)$, $q(\boldsymbol{\mu}_k)$, $q(\boldsymbol{\Lambda}_k)$, $q(\beta'_k)$, $q(\boldsymbol{\mu}'_k)$,
          $q(\boldsymbol{\Lambda}'_k)$;
    end
    for $m = 1$ to $M$ do
        Update $q(z_m)$, $q(\boldsymbol{w}_m)$, $q(z'_m)$, and $q(\boldsymbol{w}'_m)$;
    end
Until $\|q(\boldsymbol{z}^{\text{iter}}) - q(\boldsymbol{z}^{\text{iter-1}})\|_{\text{F}} \leq 1e^{-5}$ or iter=IterMAX;
return variational distributions $q(\Omega|\mathcal{D}_s)$, $q(\Omega'|\mathcal{D}_s)$ and
  random features $\{\boldsymbol{w}_m\}_{m=1}^M$, $\{\boldsymbol{w}'_m\}_{m=1}^M$.
```

TABLE I: Dataset statistics.

datasets	d	#traing examples	#test examples
ijcnn1	22	49,990	91,701
covtype	54	464,810	116,202
skin	3	122,529	122,528
EEG	14	7,490	7,490
spambase	57	2,301	2,300

Complexity: Our inference algorithm involves simple computations such as matrix addition and matrix multiplication, except that inferring $\boldsymbol{w}$ and $\boldsymbol{\Lambda}$ needs to conduct $d \times d$ matrix inversion operations, leading to $\mathcal{O}((M+T)d^3)$. Thanks to the sub-sampling scheme, based on $\mathcal{D}_s$, the total runtime per iteration is $\mathcal{O}((M+T)d^3 + MN_sT + MN_s)$. As a result, our method is quite efficient because the inference is independent of N, especially on large datasets with $N \gg d$.

V. Experiments

In this section, we experimentally evaluate the approximation performance of the proposed RFF-DIGMM model and apply it to classification tasks. All the experiments implemented in MATLAB are repeated over 10 runs on a standard PC with Intel® i5-6500 CPU (3.20 GHz) and 16 GB RAM. The source code of our implementation can be found in http://www.lfhsgre.org.

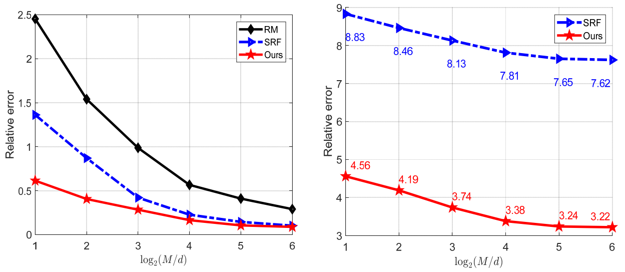
A. Experiment Setup

Datasets: We extensively study the proposed method on five large classification benchmark datasets[2] that are listed in Table I. The data in these datasets are normalized to $[0, 1]^d$ in advance, and we randomly pick half of the data for training and the rest for test on *skin*, *EEG*, and *spambase*. For *ijcnn1*, both training and test data have been divided. Following [7], we use a random 80%-20% split on *covtype*.

Kernel setting: Experiment results here are based on four non-Bochner kernels including two dot-product kernels on the unit sphere and two indefinite kernels as listed in Tab. II. These four non-Bochner kernels can be transformed into indefinite but shift-invariant kernels, and approximated by our model.

TABLE II: The used non-Bochner kernels.

Type	Kernel	Formulation
dot-product (sphere)	polynomial kernel	$\mathcal{K}_p(\boldsymbol{x}, \boldsymbol{y}) = (1 + \langle \boldsymbol{x}, \boldsymbol{y} \rangle)^p$
	Hellinger's kernel [23]	$\mathcal{K}_h(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{\langle \boldsymbol{x}, \boldsymbol{y} \rangle}$
indefinite	TL1 kernel [25]	$\mathcal{K}_\tau(\boldsymbol{x}, \boldsymbol{y}) = \max\{\tau - \|\boldsymbol{x} - \boldsymbol{y}\|_1, 0\}$
	$\tanh$ kernel [24]	$\mathcal{K}_\upsilon(\boldsymbol{x}, \boldsymbol{y}) = tanh(1 + \upsilon \langle \boldsymbol{x}, \boldsymbol{y} \rangle)$



(a) Polynomial kernel (b) TL1 kernel

Fig. 2: Comparison of RMSE on *EEG* with (a) the polynomial kernel; (b) TL1 kernel.

Parameter setting: In our experiment, the sketch size is set to $N_s = 5$. The truncation parameter in DP is $T = 5$. The order in the polynomial kernel $\mathcal{K}_p(\boldsymbol{x}, \boldsymbol{y})$ is fixed with $p = 10$, and the parameters in the TL1 kernel and $\tanh$ kernel are set to $\tau = 0.7d$ and $\upsilon = 1/d$ as suggested.

Compared methods: We choose the liblinear classifier [56] as our fast solver, and present a comparison of our method (RFF-DIGMM) with the following algorithms.

- liblinear [56]: It is an efficient solver for linear SVM. It serves as a baseline for comparison. The balance parameter C in liblinear is well tuned by 5-fold cross validation on a grid of points: $C = [2^{-5}, 2^{-4}, \ldots, 2^5]$.
- RM [38]: It adopts random Maclaurin feature maps to approximate polynomial kernels but is infeasible to the Hellinger's kernel. This is because Maclaurin expansion in RM requires the order of $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ not less than 1. Note that RM is not suited to indefinite kernel as well.
- SRF [29]: the polynomial kernel on the unit spherical is approximated by a Gaussian mixture model with ten components. Parameters in GMM are offline optimized by grid-search over $[0, 2]$.

B. Quality of Kernel Approximation

One target of the experiment is to study the approximation quality of non-Bochner kernels. In our experiment, we choose the polynomial kernel on the unit sphere and the TL1 kernel as examples. For them, we compute the groundtruth kernel matrix $\boldsymbol{K}^*$ and the approximated kernel matrix $\boldsymbol{K}$ on the *EEG* dataset, and validate the approximation quality of competing methods. The used evaluation metric here is root mean square error (RMSE) between $\boldsymbol{K}^*$ and $\boldsymbol{K}$ over N observations, i.e., RMSE $= \sqrt{\frac{1}{N(N-1)} \sum_{i=1}^N \sum_{j=1, j\neq i}^N \left(K^*_{ij} - K_{ij}\right)^2}$.

[2] All the datasets can be downloaded from https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/ or the UCI Machine Learning Repository [55].

Fig. 2 shows the kernel approximation performance of the compared algorithms with the polynomial kernel and the TL1 kernel on *EEG*. It can be observed that, in terms of polynomial kernel approximation, under varying random feature dimensionality, our method always provides lower RMSE than RM and SRF, especially when using lower dimensional random features. For TL1 kernel approximation, along with the number of random features increases, the approximation error provided by SRF and our method steadily declines. Nevertheless, SRF yields a considerable approximation error and relatively large variance. Unlike SRF, our method achieves lower RMSE, which benefits from the high flexibility of the proposed RFF-DIGMM model. Notice that, the obtained RMSE on the TL1 kernel of both two methods is not as good as those for the polynomial kernel. This is mainly due to the non-smoothness of the TL1 kernel, which enhances the approximation difficulty.

C. Classification Results for Approximating Indefinite Kernels

The main focus of our RFF-DIGMM model is not limited to improve the quality of kernel approximation. Instead, we aim to train a linear classifier in the feature space spanned by the obtained random features for classification tasks.

For the polynomial kernel, we compare the performance of random feature mappings (RM, SRF, and our method) with the polynomial kernel and the liblinear method. For the Hellinger's kernel $\mathcal{K}_h(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{\langle \boldsymbol{x}, \boldsymbol{y} \rangle}$, SRF and our RFF-DIGMM method are taken into comparisons but RM is not suited to this kernel. This is because Maclaurin expansion in RM requires the order of $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ not less than 1. Table III reports the classification accuracy and approximation time of all the competing methods for the polynomial kernel and the Hellinger's kernel. As expected, the test accuracy improves with higher-dimensional feature maps. The kernel approximation time linearly increases as the number of random features dimensionality raises. Among all the five datasets, our method achieves the best test accuracy. As a full Bayesian model, our RFF-DIGMM achieves comparable computational efficiency, and accordingly decreases the computational cost for hyper-parameter tuning and multiple trials for determining a proper number of components by SRF.

Apart from dot-product kernels, we also evaluate the classification performance of our model with the TL1 kernel and $\tanh$ kernel. The compared two algorithms include SRF and liblinear. The experimental results with respect to the test accuracy and training time are reported in Table IV. We can find that the proposed RFF-DIGMM model is superior to SRF in all the cases except for $M = 2d$ on *ijcnn1* and *spambase*. For the TL1 kernel, the test accuracy of SRF is inferior to our method, and it almost stays unchanged with nearly indiscernible improvements on *ijcnn1* and *covtype* even if M varies from $2d$ to $32d$. This phenomenon also appears to SRF with the $\tanh$ kernel on *spambase*. Instead, our RFF-DIGMM model flexibly exploits the infinite components that adapts to data, and accordingly achieving promising test accuracy on these five datasets with varying M. The classification results on two indefinite kernels demonstrate the superiority of our RFF-DIGMM model.

D. Compared with Other Kernel Approximation Methods with Bochner Kernels

As aforementioned, research works on approximating non-Bochner kernels by random features appear to be quite rare. Albeit this, we also compare the proposed RFF-DIGMM model with other recent kernel approximation algorithms as follows.

- RF [12]: It is a nonparametric kernel learning framework by learning from optimal random features.
- CROiclassification [57]: A new CRO (Concomitant Rank Order) kernel is proposed to approximate the Gaussian kernel on the unit sphere by random features.

The used kernel in CROiclassification [57] is a Gaussian kernel. Instead, as a data-dependent method, RF considers the learned random features for kernel learning and approximation. In the current setting, our method, equipped with the polynomial kernel and $M = 32d$, is taken into consideration. For the subsequent classification, all of these three algorithms are combined with the liblinear algorithm for fair comparison. The corresponding classification results are reported in Tab. V. We find that RF appears to obtain a not very promising performance even if the kernel is learned instead of directly specified. Compared to [57] equipped with the Gaussian kernel, our method with the polynomial kernel achieves a comparable classification performance and computational cost. Actually, in this paper we do not want to claim that our RFF-DIGMM model is better than other kernel approximation methods, as the scope of their applications are not the same. Instead, our aim is to show that our RFF-DIGMM model provides a justification to conduct random features for non-Bochner kernels.

E. Parametric Analysis

Here we study the influence of different sizes of the sketch, different truncation parameter, and different eigenvalue decompositions on the final results.

1) The size of the sketch: In our RFF-DIGMM model, in each iteration, we sample N_s data points from $\mathcal{D}$ for variational inference. Here we quantitatively study the influence of the size of the sketch, i.e., $N_s = 1, 5, 10, 50, 100$ in our method with the polynomial kernel and TL1 kernel on the *ijcnn1* dataset.

Fig. 3 shows the kernel approximation error, test accuracy, and time cost for polynomial kernel approximation and TL1 kernel approximation varying with different sizes of the sketch. We can see that if more data points are sampled, our method with the polynomial kernel achieves with slight improvements on the kernel approximation error and the test accuracy, see in Fig. 3(a) and Fig. 3(b). However, in terms of computational cost, the training time significantly increases along with more sampled data taken into consideration as shown in Fig. 3(c). In addition, Fig. 3(d), 3(e), 3(f) show that our model with the TL1 kernel achieves the same tendencies with the polynomial kernel setting, in terms of approximation error, classification accuracy, and time cost.

From the above experimental results, although the sketch with larger size lead to better approximation performance to some extent, this strategy cannot guarantee better classification performance. This might be because the original kernel might not be suitable for the task, as discussed in [29], [37].

TABLE III: Comparison results of various algorithms with the polynomial kernel and the Hellinger's kernel for varying feature map dimensionality (M) in terms of classification accuracy (mean±std. deviation %) and training time (mean±std. deviation sec.). The best performance is highlighted in **bold**.

	Dataset	Method	$M = 2d$	$M = 8d$	$M = 16d$	$M = 32d$	liblinear
			Acc:% (time:sec.)	Acc:% (time:sec.)	Acc:% (time:sec.)	Acc:% (time:sec.)	Acc:%
Polynomial kernel	*ijcnn1*(d=22)	RM	**90.4**±0.4 (0.3±0.0)	93.3±0.3 (1.0±0.1)	95.1±0.7 (2.4±0.2)	96.5±0.4 (5.0±0.2)	92.5±0.0
		SRF[1]	81.2±1.9 (0.3±0.0)	86.6±0.7 (1.1±0.1)	88.8±0.7 (2.2±0.1)	93.4±0.5 (4.6±0.1)	
		Ours	90.0±0.6 (0.7±0.4)	**95.7**±0.4 (2.1±0.8)	**97.3**±0.4 (3.9±0.9)	**97.8**±0.1 (10.7±0.9)	
	covtype(d=54)	RM	72.9±0.6 (5.5±0.3)	75.9±0.3 (21.8±0.5)	77.8±0.1 (44.1±1.3)	79.3±0.3 (84.3±3.2)	75.6±0.2
		SRF	68.8±0.8 (3.7±0.1)	77.3±0.3 (16.2±0.5)	80.4±0.2 (49.5±5.4)	81.9±0.2 (93.6±8.4)	
		Ours	**73.7**±0.5 (1.7±0.6)	**79.5**±0.3 (3.0±1.1)	**80.9**±0.2 (6.0±0.4)	**83.1**±0.3 (19.8±2.2)	
	skin(d=3)	RM	80.9±6.8 (0.1±0.0)	87.9±8.2 (0.2±0.0)	87.5±7.3 (0.3±0.1)	87.7±7.7 (0.6±0.0)	91.1±0.1
		SRF	91.9±2.3 (0.1±0.0)	97.8±0.1 (0.3±0.0)	98.0±0.1 (0.4±0.0)	98.1±0.1 (0.9±0.0)	
		Ours	**98.0**±0.5 (0.5±0.2)	**98.1**±0.1 (0.7±0.3)	**98.2**±0.0 (1.3±0.6)	**98.2**±0.1 (3.1±1.3)	
	EEG(d=14)	RM	64.1±1.7 (0.0±0.0)	70.3±5.0 (0.1±0.0)	74.8±2.8 (0.1±0.0)	77.8±3.8 (0.1±0.0)	63.8±0.2
		SRF	66.3±0.8 (0.0±0.0)	67.9±0.7 (0.1±0.0)	71.8±1.8 (0.1±0.0)	74.4±2.1 (0.2±0.0)	
		Ours	**68.7**±1.2 (0.3±0.2)	**80.9**±0.5 (0.9±0.5)	**83.9**±0.5 (1.7±0.8)	**85.1**±0.5 (4.0±0.8)	
	spambase(d=57)	RM	**84.6**±1.4 (0.0±0.0)	**87.5**±2.2 (0.1±0.0)	87.1±3.3 (0.2±0.0)	90.2±0.5 (0.3±0.0)	90.7±0.8
		SRF	72.4±0.8 (0.0±0.0)	80.5±2.3 (0.1±0.0)	83.2±0.4 (0.2±0.0)	84.1±0.3 (0.4±0.0)	
		Ours	79.9±0.2 (0.5±0.2)	86.7±1.2 (1.7±0.3)	**88.4**±0.9 (3.8±0.2)	**90.9**±0.5 (6.2±0.2)	
Hellinger's kernel	*ijcnn1*(d=22)	SRF	87.1±0.8 (0.3±0.0)	91.4±0.6 (1.1±0.0)	94.3±0.4 (2.2±0.0)	96.7±0.2 (4.5±0.1)	92.5±0.0
		Ours	**89.5**±0.8 (1.2±0.3)	**95.0**±1.6 (5.8±1.5)	**97.0**±0.4 (10.4±3.2)	**97.8**±0.5 (27.6±6.7)	
	covtype(d=54)	SRF	72.0±0.7 (3.5±0.1)	78.7±0.1 (14.7±0.6)	80.1±0.3 (29.8±2.4)	82.4±0.5 (58.4±8.7)	75.6±0.2
		Ours	**74.6**±0.7 (1.9±0.6)	**79.7**±0.2 (5.5±0.5)	**81.3**±0.4 (12.5±0.4)	**83.0**±0.5 (27.5±2.7)	
	skin(d=3)	SRF	96.0±1.9 (0.1±0.0)	97.9±0.2 (0.2±0.0)	98.0±0.1 (0.5±0.0)	98.1±0.1 (0.9±0.0)	91.1±0.1
		Ours	**98.3**±0.3 (0.6±0.2)	**98.2**±0.1 (1.6±0.8)	**98.2**±0.0 (3.8±1.2)	**98.2**±0.1 (6.7±2.6)	
	EEG(d=14)	SRF	65.3±2.4 (0.0±0.0)	75.5±0.7 (0.1±0.0)	82.7±0.9 (0.2±0.0)	84.2±0.5 (0.4±0.1)	63.8±0.2
		Ours	**69.3**±1.6 (0.3±0.1)	**81.0**±1.3 (1.0±0.6)	**83.9**±0.7 (1.8±0.7)	**84.3**±0.8 (4.7±1.3)	
	spambase(d=57)	SRF	75.3±1.7(0.0±0.0)	78.4±1.3 (0.1±0.0)	81.2±0.5 (0.2±0.0)	83.3±1.1 (0.4±0.1)	90.7±0.8
		Ours	**78.4**±1.2 (1.5±0.4)	**84.6**±1.0 (3.9±1.5)	**87.6**±0.8 (9.2±3.2)	**88.2**±0.2 (12.9±4.1)	

[1] For each dataset, SRF obtains parameters in GMM by an off-line grid search scheme in advance, of which the time cost is reported as follows.

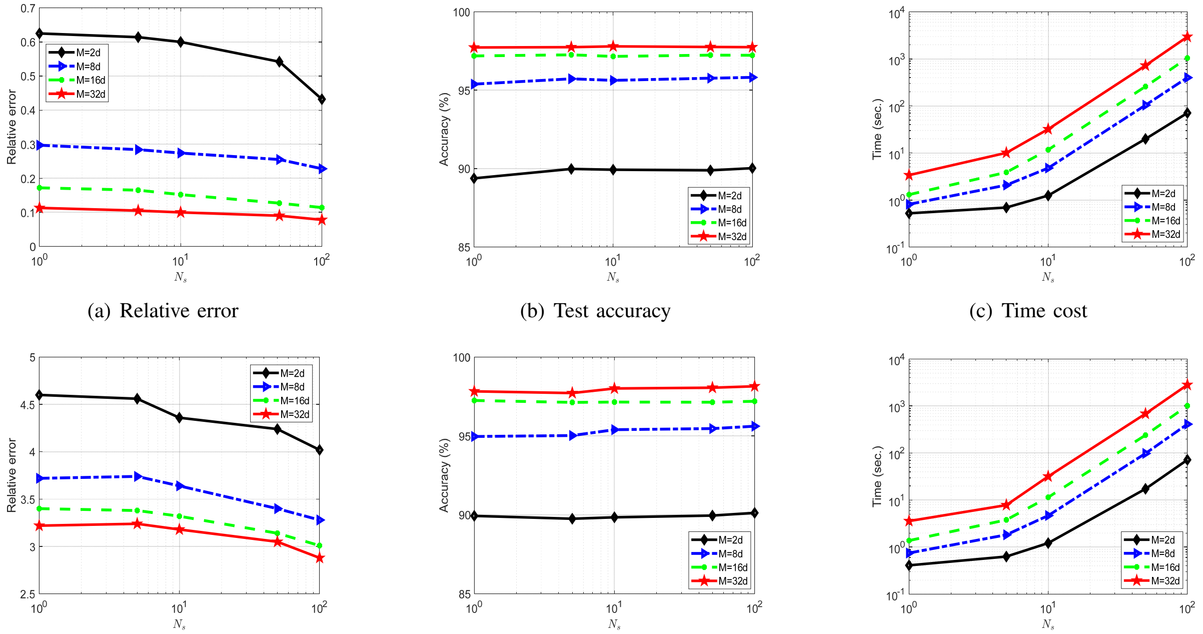
	ijcnn1	*covtype*	*skin*	*EEG*	*spambase*
the polynomial kernel (sec.)	16.7s	86.1s	18.1s	6.8s	41.3s
the Hellinger's kernel (sec.)	28.8s	21.9s	19.3s	16.2s	20.4s

TABLE IV: Comparison results of various algorithms with the TL1 kernel and the hyperbolic tangent kernel for varying feature map dimensionality (M) in terms of classification accuracy (mean±std. deviation %) and training time (mean±std. deviation sec.). The best performance is highlighted in **bold**.

	Dataset	Method	$M = 2d$	$M = 8d$	$M = 16d$	$M = 32d$	liblinear
			Acc:% (time:sec.)	Acc:% (time:sec.)	Acc:% (time:sec.)	Acc:% (time:sec.)	Acc:%
TL1 kernel	*ijcnn1*(d=22)	SRF[1]	**91.0**±0.7 (0.3±0.0)	92.0±0.2 (1.0±0.0)	92.2±0.5 (2.0±0.1)	92.1±0.2 (4.3±0.4)	92.5±0.0
		Ours	89.8±0.5 (0.6±0.4)	**95.0**±0.4 (1.8±0.9)	**97.1**±0.3 (3.8±0.8)	**97.4**±0.3 (7.8±0.7)	
	covtype(d=54)	SRF	72.7±0.7 (3.3±0.1)	73.6±0.3 (16.0±0.5)	73.7±0.2 (27.7±2.1)	73.8±0.4 (41.8±14.6)	75.6±0.2
		Ours	**73.5**±0.6 (1.9±0.6)	**86.7**±0.4 (6.3±0.5)	**87.2**±0.2 (10.8±0.3)	**87.2**±0.6 (20.0±1.2)	
	skin(d=3)	SRF	95.4±1.7 (0.1±0.0)	97.9±0.2 (0.2±0.0)	98.1±0.2 (0.4±0.0)	98.1±0.1 (0.8±0.0)	91.1±0.1
		Ours	**98.0**±0.5 (0.3±0.2)	**98.1**±0.0 (0.7±0.3)	**98.1**±0.1 (1.8±0.6)	**98.2**±0.0 (2.4±0.7)	
	EEG(d=14)	SRF	66.8±2.6 (0.1±0.0)	77.7±0.9 (0.1±0.0)	84.6±1.0(0.2±0.0)	**89.8**±0.5 (0.4±0.0)	63.8±0.2
		Ours	**69.0**±1.0 (0.3±0.2)	**80.6**±1.0 (0.9±0.6)	**85.2**±0.3 (1.7±0.8)	86.9±0.8 (3.9±0.8)	
	spambase(d=57)	SRF	**84.1**±2.4(0.0±0.0)	86.0±1.1 (0.1±0.0)	87.0±0.7 (0.2±0.0)	88.3±1.0 (0.4±0.1)	90.7±0.8
		Ours	79.9±1.0 (0.9±0.6)	**86.7**±1.1 (2.8±1.0)	**88.0**±0.9 (5.5±1.0)	**88.9**±0.6 (9.8±0.9)	
tanh kernel	*ijcnn1*(d=22)	SRF	**92.1**±0.4 (0.3±0.0)	93.0±0.4 (1.1±0.0)	93.9±0.4 (2.2±0.1)	95.5±0.8 (4.2±0.2)	92.5±0.0
		Ours	90.1±0.3 (0.6±0.4)	**95.6**±0.4 (1.9±0.9)	**97.5**±0.4 (3.7±0.9)	**97.9**±0.2 (10.4±1.2)	
	covtype(d=54)	SRF	75.8±0.8 (3.3±0.1)	78.9±0.1 (13.8±0.6)	80.4±0.3 (28.7±2.8)	82.8±0.6 (44.7±12.1)	75.6±0.2
		Ours	**74.2**±1.2 (1.7±0.5)	**79.6**±0.1 (4.9±0.7)	**81.0**±0.2 (10.4±0.5)	**83.6**±0.5 (22.5±1.5)	
	skin(d=3)	SRF	94.2±2.3 (0.1±0.0)	**98.2**±0.2 (0.2±0.0)	**98.2**±0.1 (0.4±0.0)	98.1±0.1 (0.8±0.0)	91.1±0.1
		Ours	**97.2**±1.5 (0.6±0.2)	**98.2**±0.1 (1.4±0.5)	**98.2**±0.0 (2.1±0.8)	**98.2**±0.1 (4.2±0.8)	
	EEG(d=14)	SRF	64.3±1.2 (0.1±0.0)	75.2±1.1 (0.2±0.0)	78.4±0.8 (0.3±0.0)	79.1±0.9 (0.5±0.1)	63.8±0.2
		Ours	**69.3**±1.2 (0.3±0.2)	**80.2**±1.0 (1.9±0.8)	**83.4**±0.5 (1.7±0.8)	**84.5**±1.2 (4.0±0.7)	
	spambase(d=57)	SRF	**83.5**±1.8(0.1±0.0)	84.2±1.1 (0.1±0.0)	84.8±0.8 (0.2±0.0)	85.0±1.4 (0.4±0.1)	90.7±0.8
		Ours	76.2±1.4 (2.0±0.7)	**85.6**±1.5 (4.8±1.0)	**87.0**±0.8 (7.5±2.5)	**87.2**±1.2 (20.7±4.8)	

[1] For each dataset, SRF obtains parameters in GMM by an off-line grid search scheme in advance, of which the time cost is reported as follows.

	ijcnn1	*covtype*	*skin*	*EEG*	*spambase*
the TL1 kernel (sec.)	3.7s	8.3s	10.7s	3.0s	6.5s
the $\tanh$ kernel (sec.)	10.2s	11.8s	29.5s	16.5s	17.4s



(a) Relative error

(b) Test accuracy

(c) Time cost

(d) Relative error

(e) Test accuracy

(f) Time cost

Fig. 3: Comparison of results versus varying N_s for the polynomial kernel (a), (b), (c) and the TL1 kernel (d), (e), (f) on the *ijcnn1* dataset.

TABLE V: Comparison results of various representative algorithms with Bochner kernels and our RFF-DIGMM model with the polynomial kernel. The best scores are highlighted by **bold**.

Dataset	RF	CROiclassification	RFF-DIGMM
	learned kernel	Gaussian kernel	polynomial kernel
	Acc:% (time:sec.)	Acc:% (time:sec.)	Acc:% (time:sec.)
ijcnn1	95.5±0.4 (36.6±2.3)	97.1±0.1 (0.4±0.1)	**97.8**±0.1 (10.7±0.9)
covtype	79.2±0.28 (44.4±2.2)	**86.2**±0.2 (17.7±0.9)	83.1±0.3 (19.8±2.2)
skin	97.9±0.2 (40.4±1.4)	98.1±0.1 (3.4±0.3)	**98.2**±0.1 (3.1±1.3)
EEG	84.3±0.8 (5.0±0.1)	**87.2**±0.1 (0.4±0.1)	85.1±0.5 (4.0±0.8)
spambase	86.2±1.4 (2.0±0.2)	90.2±0.1 (0.2±0.1)	**90.9**±0.5 (6.2±0.2)

2) Truncation Parameter: As aforementioned, the variational distribution is truncated but our model is a full DP and is not truncated. The truncation level T is a variational parameter which can be freely set; it is not a part of the prior model specification. Here we evaluate the parametric sensitivity of T on the *ijcnn1* dataset. Table VI reports the classification accuracy and time cost for computing random features when T is chosen as 1, 5, and 10. It can be observed that the test accuracy with different T is experimentally stable. However, the time cost gradually increases as T rises. Hence, small T values are shown to achieve high computational efficiency, which explains the reason why we choose this parameter for our experiments.

F. Illustration of Convergence

Here we investigate the convergence of the used non-conjugate variational inference algorithm. We take the TL1 kernel with $M = 2d$ as an example, and plot $\|q(\boldsymbol{z}^t) - q(\boldsymbol{z}^{(t-1)})\|_{\mathrm{F}}$ versus iteration on the above-mentioned five datasets in Fig. 4. It can be found that, in most cases, $q(\boldsymbol{z}^t)$ significantly decays in the first 5 iterations in our variational inference algorithm, which leads to quick convergence under the stopping criterion $\|q(\boldsymbol{z}^t) - q(\boldsymbol{z}^{(t-1)})\|_{\mathrm{F}} \leq 1e^{-5}$. The total iterations are less than 10 in these five datasets except for *skin* with about 13 iterations. Therefore, the maximum iteration number fixed to 50 is reasonable and enough. And further, the convergence of the optimization process employed by our non-conjugate variational inference is well demonstrated.

VI. Conclusion

We investigated a full non-parametric Bayesian method in random feature mappings for indefinite kernels. It extends the traditional Bochner kernel in RFF to several non-Bochner kernels including dot-product kernels and indefinite kernels. By placing a DP prior on the components of Gaussian mixtures, our RFF-DIGMM model is adaptive to the data with varying components. The derived non-conjugate variational inference algorithm with the sub-sampling scheme is efficient and effective for model inference. As a result, the superiority of our method is demonstrated by experimental validation on several classification datasets.

Appendix A
Update variational factors

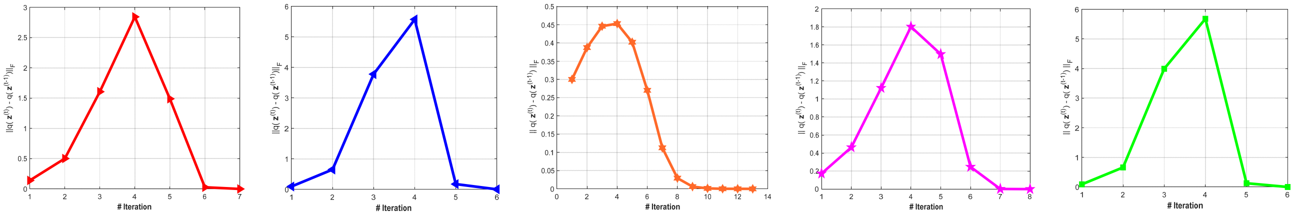
The optimization for each variational factor is conducted by iteratively updating latent variables in details.

1) $q(\beta_t)$: We absorb terms in Eq. (13) that do not depend on β_t into the additive normalization constant, giving

$$\begin{aligned}\ln q^*(\beta_t) &= \mathbb{E}_{\Omega\setminus\beta_t} \ln p(\mathcal{D}_s, \Omega) + \text{const} \\ &= \ln p(\beta_t) + \sum_{m=1}^{M} \mathbb{E}_q \Big[\ln p(z_m|\tilde{\boldsymbol{\beta}}) \Big] + c\,.\end{aligned}$$

TABLE VI: Comparison results of different truncation parameter values on the *ijcnn1* dataset.

kernel type	T	$M=2d$	$M=4d$	$M=8d$	$M=16d$	$M=32d$
		Acc:% (time:sec.)	Acc:% (time:sec.)	Acc:% (time:sec.)	Acc:% (time:sec.)	Acc:% (time:sec.)
Polynomial kernel	$T=1$	89.9±0.6 (0.2±0.2)	92.8±0.9 (0.4±0.2)	95.7±0.4 (0.6±0.2)	97.0±0.4 (1.0±0.2)	98.1±0.3 (2.4±0.2)
	$T=5$	90.0±0.6 (0.7±0.4)	92.1±0.7 (1.1±0.6)	95.7±0.4 (2.1±0.8)	97.3±0.4 (3.9±0.9)	97.8±0.1 (10.7±0.9)
	$T=10$	89.7±0.2 (0.8±0.4)	91.9±0.8 (1.1±0.7)	94.8±0.7 (2.7±0.9)	97.4±0.4 (5.3±0.7)	98.2±0.2 (12.0±0.5)
TL1 kernel	$T=1$	90.1±0.7 (0.2±0.2)	92.2±0.8 (0.3±0.2)	95.3±0.5 (0.5±0.2)	97.0±0.2 (0.8±0.4)	97.8±0.3 (1.3±0.6)
	$T=5$	89.8±0.5 (0.6±0.4)	91.7±0.6 (1.0±0.5)	95.0±0.4 (1.8±0.9)	97.1±0.3 (3.8±0.8)	97.4±0.3 (7.8±0.7)
	$T=10$	89.9±0.3 (0.7±0.4)	92.0±0.7 (1.1±0.7)	94.9±0.5 (2.0±0.9)	96.9±0.3 (4.1±0.8)	97.8±0.2 (11.8±0.7)



(a) *ijcnn1* (b) *covtype* (c) *skin* (d) *EEG* (e) *spambase*

Fig. 4: Convergence plots on (a) *ijcnn1*, (b) *covtype*, (c) *skin*, (d) *EEG*, and (e) *spambase*.

Following [52] and $q(z_m > T) = 0$, we have

$$\mathbb{E}_q\Big[\ln p(z_m|\tilde{\boldsymbol{\beta}})\Big]{=}\sum_{k=1}^{T}\Big\{q(z_m{>}k)\mathbb{E}_q\big[\ln(1{-}\beta_k)\big]+q(z_m{=}k)\mathbb{E}(\ln\beta_k)\Big\}\,.$$

As a result, the optimal variational distribution $q^*(\beta_t)$ can be obtained by

$$\begin{aligned}\ln q^*(\beta_t) &{=}\ln p(\beta_t){+}\sum_{m=1}^{M}[q(z_m{>}t)\ln(1{-}\beta_t){+}q(z_m{=}t)\ln\beta_t]{+}c\\ &{=}\ln p(\beta_t){+}\Bigg[\sum_{m=1}^{M}q(z_m{>}t)\Bigg]\ln(1-\beta_t){+}\Bigg[\sum_{m=1}^{M}q(z_m{=}t)\Bigg]\ln\beta_t{+}c\,.\end{aligned}$$

Since $\beta_k \sim \text{Beta}(1,\alpha_0)$, we have $p(\beta_k) \propto (1-\beta_k)^{\alpha_0-1}$. Finally, we have

$$\beta_t \sim \text{Beta}\Bigg(1+\sum_{m=1}^{M}q(z_m=t),\alpha_0+\sum_{m=1}^{M}q(z_m>t)\Bigg)\,.$$

2) $q(z_m)$: Likewise, we do not consider irrelevant terms of z_m in Eq. (13), i.e.

$$\begin{aligned}\ln q^*(z_m) &= \mathbb{E}_{\Omega\backslash z_m}\ln p(\mathcal{D}_s,\Omega)+c\\ &= \mathbb{E}_q\Big[\ln p(z_m|\tilde{\boldsymbol{\beta}})+\ln p(\boldsymbol{w}_m|z_m,\tilde{\boldsymbol{\mu}},\tilde{\boldsymbol{\Lambda}})\Big]+c\\ &= \sum_{k=1}^{T}\Big\{1[z_m>k]\mathbb{E}[\ln(1-\beta_k)]+1[z_m=k]\mathbb{E}(\ln\beta_k)\\ &\quad+1(z_m=k)\Big(\frac{1}{2}\mathbb{E}\big[\ln|\boldsymbol{\Lambda}_k|\big]-\frac{d}{2}\ln(2\pi)-\frac{1}{2}\Xi\Big)\Big\}+c\,,\end{aligned}$$

with $\Xi \triangleq \mathbb{E}_{\boldsymbol{w}_m,\boldsymbol{\mu}_k,\boldsymbol{\Lambda}_k}\big[(\boldsymbol{w}_m-\boldsymbol{\mu}_k)^\top\boldsymbol{\Lambda}_k(\boldsymbol{w}_m-\boldsymbol{\mu}_k)\big]$. Defining

$$\ln\hbar_{mk}{=}\mathbb{E}(\ln\beta_k){+}\sum_{t=1}^{k-1}\mathbb{E}[\ln(1{-}\beta_t)]{+}\frac{1}{2}\Big(\mathbb{E}\ln|\boldsymbol{\Lambda}_k|{-}d\ln(2\pi){-}\Xi\Big)\,,$$

and scaling $\tilde{\hbar}_{mk}=\frac{\hbar_{mk}}{\sum_{t=1}^{T}\hbar_{mt}}$, we have $q(z_m=k)=\tilde{\hbar}_{mk}$. It means that z_m is chosen according to a multinomial probability distribution.

3) $q(\boldsymbol{\mu}_k)$: Keeping only terms that have a functional dependence on $\boldsymbol{\mu}_k$, we have

$$\begin{aligned}\ln q^*(\boldsymbol{\mu}_k) &= \mathbb{E}_{\Omega\backslash\boldsymbol{\mu}_k}\ln p(\mathcal{D}_s,\Omega)+c\\ &= \mathbb{E}_q\Big[\ln p(\boldsymbol{\mu}_k)+\ln\prod_{m=1}^{M}p(\boldsymbol{w}_m|z_m,\tilde{\boldsymbol{\mu}},\tilde{\boldsymbol{\Lambda}})\Big]+c\\ &= \ln p(\boldsymbol{\mu}_k){+}\sum_{m=1}^{M}q(z_m{=}k)\mathbb{E}_{\Omega\backslash\boldsymbol{\mu}_k}\big[\ln\mathcal{N}(\boldsymbol{w}_m|\boldsymbol{\mu}_k,\boldsymbol{\Lambda}_k^{-1})\big]{+}c\\ &= -\frac{1}{2}\boldsymbol{\mu}_k^\top\Big(\boldsymbol{R}_0+\mathbb{E}(\boldsymbol{\Lambda}_k)\sum_{m=1}^{M}q(z_m=k)\Big)\boldsymbol{\mu}_k\\ &\quad+\boldsymbol{\mu}_k^\top\Big(\boldsymbol{R}_0\boldsymbol{m}_0+\mathbb{E}(\boldsymbol{\Lambda}_k)\sum_{m=1}^{M}q(z_m=k)\mathbb{E}(\boldsymbol{w}_m)\Big)\,.\end{aligned}$$

After some algebraic manipulations, as we expect, $\boldsymbol{\mu}_k$ is subject to a Gaussian distribution $\boldsymbol{\mu}_k \sim \mathcal{N}\big(\boldsymbol{\mu}_k|\boldsymbol{m}_k,\boldsymbol{R}_k^{-1}\big)$ with the following mean vector and precision matrix

$$\left\{\begin{aligned}\boldsymbol{m}_k &= \boldsymbol{R}_k^{-1}\Big(\boldsymbol{R}_0\boldsymbol{m}_0+\mathbb{E}(\boldsymbol{\Lambda}_k)\sum_{m=1}^{M}q(z_m=k)\mathbb{E}(\boldsymbol{w}_m)\Big)\\ \boldsymbol{R}_k &= \boldsymbol{R}_0+\mathbb{E}(\boldsymbol{\Lambda}_k)\sum_{m=1}^{M}q(z_m=k)\,.\end{aligned}\right.$$

4) $q(\boldsymbol{\Lambda}_k)$: We only retain some terms with respect to $\boldsymbol{\Lambda}_k$ in Eq. (13), namely

$$
\begin{aligned}
&\ln q^*(\boldsymbol{\Lambda}_k) = \mathbb{E}_{\Omega\backslash\boldsymbol{\Lambda}_k} \ln p(\mathcal{D}_s, \Omega) + c \\
&= \ln p(\boldsymbol{\Lambda}_k) + \sum_{m=1}^{M} q(z_m = k)\mathbb{E}_{\Omega\backslash\boldsymbol{\Lambda}_k}\big[\ln\mathcal{N}(\boldsymbol{w}_m|\boldsymbol{\mu}_k, \boldsymbol{\Lambda}_k^{-1})\big] + c \\
&= -\frac{1}{2}\mathrm{Tr}(\boldsymbol{\Lambda}_k\boldsymbol{W}_0^{-1}) + \frac{\nu_0 - d - 1}{2}\ln|\boldsymbol{\Lambda}_k| + \frac{1}{2}\Bigg(\sum_{m=1}^{M} q(z_m = k)\Bigg)\ln|\boldsymbol{\Lambda}_k| \\
&\quad -\frac{1}{2}\sum_{m=1}^{M} q(z_m = k)\mathrm{Tr}\Big(\boldsymbol{\Lambda}_k\mathbb{E}(\boldsymbol{w}_m - \boldsymbol{\mu}_k)(\boldsymbol{w}_m - \boldsymbol{\mu}_k)^\top\Big) + c\,.
\end{aligned}
$$

Thus we have $\boldsymbol{\Lambda}_k \sim \mathcal{W}(\boldsymbol{\Lambda}_k|\boldsymbol{W}_k, \nu_k)$ with $\nu_k = \nu_0 + \sum_{m=1}^{M} q(z_m = k)$ and $\boldsymbol{W}_k^{-1}$ is formulated by

$$
\boldsymbol{W}_k^{-1} = \boldsymbol{W}_0^{-1} + \sum_{m=1}^{M} q(z_m = k)\mathbb{E}(\boldsymbol{w}_m - \boldsymbol{\mu}_k)(\boldsymbol{w}_m - \boldsymbol{\mu}_k)^\top\,.
$$

5) $q(\boldsymbol{w}_m)$: Inspecting Eq. (13) and reading off those terms which involve only $\boldsymbol{w}_m$, we have

$$
\begin{aligned}
&\ln q^*(\boldsymbol{w}_m) = \mathbb{E}_{\Omega\backslash\boldsymbol{w}_m} \ln p(\mathcal{D}_s, \Omega) + c \\
&= \mathbb{E}_q\big[\ln p(\boldsymbol{w}_m|z_m, \tilde{\boldsymbol{\mu}}, \tilde{\boldsymbol{\Lambda}})\big] + \mathbb{E}_q\Big[\ln\prod_{i,j=1,i\neq j}^{N_s} p(K_{ij}^{+}|(\boldsymbol{x}_i, \boldsymbol{x}_j), \boldsymbol{w}_m)\Big] + c\,.
\end{aligned}
$$

For the first term, we have

$$
\begin{aligned}
&\mathbb{E}_q\big[\ln p(\boldsymbol{w}_m|z_m, \tilde{\boldsymbol{\mu}}, \tilde{\boldsymbol{\Lambda}})\big] = \mathbb{E}_q\big[\ln p(\boldsymbol{w}_m|\tilde{\boldsymbol{\mu}}, \tilde{\boldsymbol{\Lambda}})^{1[z_m = k]}\big] \\
&= \frac{1}{2}\sum_{k=1}^{T} q(z_m = k)\Big(\mathbb{E}\big[\ln|\boldsymbol{\Lambda}_k|\big] - \mathbb{E}_{\boldsymbol{\mu}_k,\boldsymbol{\Lambda}_k}\big[(\boldsymbol{w}_m - \boldsymbol{\mu}_k)^\top\boldsymbol{\Lambda}_k(\boldsymbol{w}_m - \boldsymbol{\mu}_k)\big]\Big) + c\,.
\end{aligned}
$$

The second term can be expressed as

$$
\begin{aligned}
&\mathbb{E}_q\Big[\ln\prod_{i,j=1,i\neq j}^{N_s} p(K_{ij}^{+}|(\boldsymbol{x}_i, \boldsymbol{x}_j), \boldsymbol{w}_m)\Big] \\
&= -\frac{1}{2\sigma_\epsilon^2}\sum_{i,j=1,i\neq j}^{N_s}\Big(K_{ij}^{+} - \cos\big[\boldsymbol{w}_m^\top(\boldsymbol{x}_i - \boldsymbol{x}_j)\big]\Big)^2 + c\,.
\end{aligned}
$$

Since $\boldsymbol{w}_m$ is not a conjugate variable, we conduct the second order Taylor expansion $\cos\big[\boldsymbol{w}_m^\top(\boldsymbol{x}_i - \boldsymbol{x}_j)\big] \approx 1 - \frac{1}{2}\boldsymbol{w}_m^\top(\boldsymbol{x}_i - \boldsymbol{x}_j)(\boldsymbol{x}_i - \boldsymbol{x}_j)^\top\boldsymbol{w}_m$, and derive that

$$
\ln q^*(\boldsymbol{w}_m) \approx -\frac{1}{2}\boldsymbol{w}_m^\top\boldsymbol{S}\boldsymbol{w}_m + \boldsymbol{w}_m^\top\sum_{k=1}^{T}\big[q(z_m = k)\mathbb{E}(\boldsymbol{\Lambda}_k)\mathbb{E}(\boldsymbol{\mu}_k)\big] + c\,.
$$

where $\boldsymbol{S}$ is defined by

$$
\boldsymbol{S} = \sum_{k=1}^{T}\big[q(z_m = k)\mathbb{E}(\boldsymbol{\Lambda}_k)\big] + \frac{1}{2\sigma_\epsilon^2}\sum_{i,j=1,i\neq j}^{N_s}(1 - K_{ij}^{+})(\boldsymbol{x}_i - \boldsymbol{x}_j)(\boldsymbol{x}_i - \boldsymbol{x}_j)^\top.
$$

Therefore, $\boldsymbol{w}_m$ is subject to

$$
\boldsymbol{w}_m \sim \mathcal{N}\Bigg(\boldsymbol{S}^{-1}\Big\{\sum_{k=1}^{T}\big[q(z_m = k)\mathbb{E}(\boldsymbol{\Lambda}_k)\mathbb{E}(\boldsymbol{\mu}_k)\big]\Big\}, \boldsymbol{S}^{-1}\Bigg)\,.
$$

In the variational update equations, we also need to calculate the expectations with respect to the current variational distributions. For example, $\mathbb{E}(\boldsymbol{\mu}_k)$ and $\mathbb{E}(\boldsymbol{\Lambda}_k)$ can be easily obtained by their respective distributions. Here we present several intractable expectation computations. The expectation $\mathbb{E}\big(\ln|\boldsymbol{\Lambda}_k|\big)$ can be obtained by

$$
\mathbb{E}\big(\ln|\boldsymbol{\Lambda}_k|\big) = \sum_{i=1}^{d}\psi\Big(\frac{\nu_k + 1 - i}{2}\Big) + d\ln 2 + \ln|\boldsymbol{W}_k|\,,
$$

where $\psi(\cdot)$ is the digamma function with $\psi(x) = \frac{\mathrm{d}}{\mathrm{d}x}\ln\Gamma(x)$. Besides, the expectation $\mathbb{E}_{\boldsymbol{\mu}_k,\boldsymbol{\Lambda}_k}\big[(\boldsymbol{w}_m - \boldsymbol{\mu}_k)^\top\boldsymbol{\Lambda}_k(\boldsymbol{w}_m - \boldsymbol{\mu}_k)\big]$ is given by

$$
\begin{aligned}
&\mathbb{E}_{\boldsymbol{\mu}_k,\boldsymbol{\Lambda}_k}\big[(\boldsymbol{w}_m - \boldsymbol{\mu}_k)^\top\boldsymbol{\Lambda}_k(\boldsymbol{w}_m - \boldsymbol{\mu}_k)\big] \\
&= \int_{\boldsymbol{\mu}_k}\int_{\boldsymbol{\Lambda}_k}\mathrm{Tr}\Big[\boldsymbol{\Lambda}_k(\boldsymbol{w}_m - \boldsymbol{\mu}_k)^\top(\boldsymbol{w}_m - \boldsymbol{\mu}_k)\Big]q^*(\boldsymbol{\mu}_k)q^*(\boldsymbol{\Lambda}_k)\mathrm{d}\boldsymbol{\mu}_k\mathrm{d}\boldsymbol{\Lambda}_k \\
&= \int_{\boldsymbol{\Lambda}_k}\int_{\boldsymbol{\mu}_k}\mathrm{Tr}\Big[\boldsymbol{\Lambda}_k(\boldsymbol{w}_m^\top\boldsymbol{w}_m - 2\boldsymbol{w}_m^\top\boldsymbol{\mu}_k + \boldsymbol{\mu}_k^\top\boldsymbol{\mu}_k)\Big]q^*(\boldsymbol{\mu}_k)\mathrm{d}\boldsymbol{\mu}_k q^*(\boldsymbol{\Lambda}_k)\mathrm{d}\boldsymbol{\Lambda}_k \\
&= \int_{\boldsymbol{\Lambda}_k}\mathrm{Tr}\Big[\boldsymbol{\Lambda}_k(\boldsymbol{w}_m^\top\boldsymbol{w}_m - 2\boldsymbol{w}_m^\top\boldsymbol{m}_k + \boldsymbol{m}_k^2 + \boldsymbol{R}_k^{-1})\Big]q^*(\boldsymbol{\Lambda}_k)\mathrm{d}\boldsymbol{\Lambda}_k \\
&= \mathbb{E}(\boldsymbol{\Lambda}_k)\Big[(\boldsymbol{w}_m - \boldsymbol{m}_k)^\top(\boldsymbol{w}_m - \boldsymbol{m}_k) + \boldsymbol{R}_k^{-1}\Big]\,.
\end{aligned}
$$

Similarly, the expectations $\mathbb{E}_{\boldsymbol{w}_m,\boldsymbol{\mu}_k,\boldsymbol{\Lambda}_k}\big[(\boldsymbol{w}_m - \boldsymbol{\mu}_k)^\top\boldsymbol{\Lambda}_k(\boldsymbol{w}_m - \boldsymbol{\mu}_k)\big]$ and $\mathbb{E}_{\boldsymbol{w}_m,\boldsymbol{\mu}_k}\big[(\boldsymbol{w}_m - \boldsymbol{\mu}_k)^\top\boldsymbol{\Lambda}_k(\boldsymbol{w}_m - \boldsymbol{\mu}_k)\big]$ can be calculated by the above way.

References

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

[2] A. G. Wilson, Z. Hu, R. Salakhutdinov, and E. P. Xing, "Deep kernel learning," in *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2016, pp. 370–378.

[3] C. M. Alaíz, M. Fanuel, and J. A. Suykens, "Convex formulation for kernel PCA and its use in semi-supervised learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 8, pp. 3863–3869, 2018.

[4] V. N. Vapnik, *The Nature of Statistical Learning Theory*. Springer, 1995.

[5] H. Drucker, C. J. Burges, L. Kaufman, A. J. Smola, and V. Vapnik, "Support vector regression machines," in *Proceedings of Advances in neural information processing systems*, 1997, pp. 155–161.

[6] B. Schölkopf, A. Smola, and K.-R. Müller, "Kernel principal component analysis," in *Proceedings of International Conference on Artificial Neural Networks*, 1997, pp. 583–588.

[7] C.-J. Hsieh, S. Si, and I. Dhillon, "A divide-and-conquer solver for kernel support vector machines," in *Proceedings of the International Conference on Machine Learning*, 2014, pp. 566–574.

[8] Y. Zhang, J. Duchi, and M. Wainwright, "Divide and conquer kernel ridge regression," in *Proceedings of Conference on Learning Theory*, 2013, pp. 592–617.

[9] A. J. Smola and B. Schölkopf, "Sparse greedy matrix approximation for machine learning," in *Proceedings of the International Conference on Machine Learning*, 2000, pp. 911–918.

[10] S. Fine and K. Scheinberg, "Efficient SVM training using low-rank kernel representations," *Journal of Machine Learning Research*, vol. 2, no. 2, pp. 243–264, 2001.

[11] C. K. Williams and M. Seeger, "Using the nyström method to speed up kernel machines," in *Proceedings of Advances in Neural Information Processing Systems*, 2001, pp. 682–688.

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

[13] Z. Li, J.-F. Ton, D. Oglic, and D. Sejdinovic, "Towards a unified analysis of random Fourier features," in *Proceedings of the 36th International Conference on Machine Learning*, 2019, pp. 3905–3914.

[14] A. Rahimi and B. Recht, "Random features for large-scale kernel machines," in *Proceedings of Advances in Neural Information Processing Systems*, 2007, pp. 1177–1184.

[15] Y. Sun, A. Gilbert, and A. Tewari, "But how does it work in theory? Linear SVM with random features," in *Proceedings of Advances in Neural Information Processing Systems*, 2018, pp. 3383–3392.

[16] K. Cutajar, E. V. Bonilla, P. Michiardi, and M. Filippone, "Random feature expansions for deep Gaussian processes," in *Proceedings of the International Conference on Machine Learning*, 2017, pp. 884–893.

[17] B. Xie, Y. Liang, and L. Song, "Scale up nonlinear component analysis with doubly stochastic gradients," in *Proceedings of Advances in Neural Information Processing Systems*, 2015, pp. 2341–2349.

[18] D. Lopez-Paz, S. Sra, A. Smola, Z. Ghahramani, and B. Schölkopf, "Randomized nonlinear component analysis," in *Proceedings of the International Conference on Machine Learning*, 2014, pp. 1359–1367.

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

[20] C. S. Ong, X. Mary, and A. J. Smola, "Learning with non-positive kernels," in *Proceedings of the International Conference on Machine Learning*, 2004, pp. 81–89.

[21] F. Liu, X. Huang, C. Gong, J. Yang, and J. A. Suykens, "Indefinite kernel logistic regression with concave-inexact-convex procedure," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 3, pp. 765–776, 2019.

[22] G. Loosli, S. Canu, and S. O. Cheng, "Learning SVM in Kreĭn spaces," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 6, pp. 1204–1216, 2016.

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

[24] A. J. Smola, Z. L. Ovari, and R. C. Williamson, "Regularization with dot-product kernels," in *Proceedings of Advances in Neural Information Processing Systems*, 2001, pp. 308–314.

[25] X. Huang, J. A. Suykens, S. Wang, J. Hornegger, and A. Maier, "Classification with truncated ℓ_1 distance kernel," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 5, pp. 2025 – 2030, 2018.

[26] A. Feragen, F. Lauze, and S. Hauberg, "Geodesic exponential kernels: when curvature and linearity conflict," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3032–3042.

[27] N. Pham and R. Pagh, "Fast and scalable polynomial kernels via explicit feature maps," in *Proceedings of ACM International Conference on Knowledge Discovery and Data Mining*, 2013, pp. 239–247.

[28] R. Hamid, Y. Xiao, A. Gittens, and D. Decoste, "Compact random feature maps," in *Proceedings of the International Conference on Machine Learning*, 2014, pp. 19–27.

[29] J. Pennington, F. X. X. Yu, and S. Kumar, "Spherical random features for polynomial kernels," in *Proceedings of Advances in Neural Information Processing Systems*, 2015, pp. 1846–1854.

[30] S. Roweis and Z. Ghahramani, "A unifying review of linear Gaussian models," *Neural Computation*, vol. 11, no. 2, pp. 305–345, 1999.

[31] J. Bognár, *Indefinite inner product spaces*. Springer, 1974.

[32] F. X. Yu, A. T. Suresh, K. Choromanski, D. Holtmannrice, and S. Kumar, "Orthogonal random features," in *Proceedings of Advances in Neural Information Processing Systems*, 2016, pp. 1975–1983.

[33] M. J. Wainwright and M. I. Jordan, "Graphical models, exponential families, and variational inference," *Foundations and Trends® in Machine Learning*, vol. 1, no. 1–2, pp. 1–305, 2008.

[34] S. J. Gershman and D. M. Blei, "A tutorial on Bayesian nonparametric models," *Journal of Mathematical Psychology*, vol. 56, no. 1, pp. 1–12, 2012.

[35] H. Avron, V. Sindhwani, J. Yang, and M. W. Mahoney, "Quasi-Monte Carlo feature maps for shift-invariant kernels," *Journal of Machine Learning Research*, vol. 17, no. 1, pp. 4096–4133, 2016.

[36] Q. Le, T. Sarlós, and A. Smola, "FastFood-approximating kernel expansions in loglinear time," in *Proceedings of the International Conference on Machine Learning*, 2013, pp. 1–9.

[37] M. Munkhoeva, Y. Kapushev, E. Burnaev, and I. Oseledets, "Quadrature-based features for kernel approximation," in *Proceedings of Advances in Neural Information Processing Systems*, 2018, pp. 9165–9174.

[38] P. Kar and H. Karnick, "Random feature maps for dot product kernels," in *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2012, pp. 583–591.

[39] T. S. Ferguson, "A Bayesian analysis of some nonparametric problems," *The Annals of Statistics*, pp. 209–230, 1973.

[40] C. E. Rasmussen, "The infinite Gaussian mixture model," in *Proceedings of Advances in Neural Information Processing Systems*, 2000, pp. 554–560.

[41] J. Sethuraman, "A constructive definition of Dirichlet priors," *Statistica sinica*, pp. 639–650, 1994.

[42] P. Orbanz and Y. W. Teh, *Bayesian Nonparametric Models*. Springer US, 2011.

[43] I. Steinwart and C. Andreas, *Support Vector Machines*. Springer Science and Business Media, 2008.

[44] Z. C. Guo and L. Shi, "Optimal rates for coefficient-based regularized regression," *Applied and Computational Harmonic Analysis*, pp. 1–40, 2017.

[45] J. B. Oliva, A. Dubey, A. G. Wilson, B. Póczos, J. Schneider, and E. P. Xing, "Bayesian nonparametric kernel learning," in *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2016, pp. 1078–1086.

[46] V. Maz'ya and G. Schmidt, "On approximate approximations using Gaussian kernels," *IMA Journal of Numerical Analysis*, vol. 16, no. 1, pp. 13–29, 1996.

[47] B. W. Silverman, *Density estimation for statistics and data analysis*. Routledge, 2018.

[48] M. D. Hoffman, D. M. Blei, C. Wang, and J. Paisley, "Stochastic variational inference," *Journal of Machine Learning Research*, vol. 14, no. 1, pp. 1303–1347, 2013.

[49] S. I. Amari, "Natural gradient works efficiently in learning," *Neural Computation*, vol. 10, no. 2, pp. 251–276, 1998.

[50] M. Luby and A. Wigderson, "Pairwise independence and derandomization," *Foundations and Trends® in Theoretical Computer Science*, vol. 1, no. 4, pp. 237–301, 2006.

[51] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational inference: A review for statisticians," *Journal of the American Statistical Association*, vol. 112, no. 518, 2016.

[52] D. M. Blei and M. I. Jordan, "Variational inference for dirichlet process mixtures," *Bayesian analysis*, vol. 1, no. 1, pp. 121–143, 2006.

[53] C. M. Bishop, *Pattern Recognition and Machine Learning*. springer, 2006.

[54] C. Wang and D. M. Blei, "Variational inference in nonconjugate models," *Journal of Machine Learning Research*, vol. 14, no. 4, pp. 1005–1031, 2013.

[55] C. Blake and C. J. Merz, "UCI Repository of Machine Learning Databases," 1998. [Online]. Available: http://archive.ics.uci.edu/ml/

[56] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "LIBLINEAR: a library for large linear classification," *Journal of Machine Learning Research*, vol. 9, pp. 1871–1874, 2008.

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