The Fast Kernel Transform

John Paul Ryan
Cornell University
johnryan@cs.cornell.edu

Sebastian Ament
Cornell University
sea79@cornell.edu

Carla P. Gomes
Cornell University
gomes@cs.cornell.edu

Anil Damle
Cornell University
damle@cornell.edu

Abstract

Kernel methods are a highly effective and widely used collection of modern machine learning algorithms. A fundamental limitation of virtually all such methods are computations involving the kernel matrix that naïvely scale quadratically (e.g., matrix-vector multiplication) or cubically (solving linear systems) with the size of the dataset N. We propose the Fast Kernel Transform (FKT), a general algorithm to compute matrix-vector multiplications (MVMs) for datasets in moderate dimensions with quasilinear complexity. Typically, analytically grounded fast multiplication methods require specialized development for specific kernels. In contrast, our scheme is based on auto-differentiation and automated symbolic computations that leverage the analytical structure of the underlying kernel. This allows the FKT to be easily applied to a broad class of kernels, including Gaussian, Matérn, and Rational Quadratic covariance functions and Green's functions, including those of the Laplace and Helmholtz equations. Furthermore, the FKT maintains a high, quantifiable, and controllable level of accuracy—properties that many acceleration methods lack. We illustrate the efficacy and versatility of the FKT by providing timing and accuracy benchmarks with comparisons to adjacent methods, and by applying it to scale the stochastic neighborhood embedding (t-SNE) and Gaussian processes to large real-world datasets.

Proceedings of the 25[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2022, Valencia, Spain. PMLR: Volume 151.

1 INTRODUCTION

Kernel methods are fundamental to machine learning and many of its applications. Examples include kernel density estimation, kernel regression, Gaussian processes, support vector machines, kernel clustering, and kernel PCA (Shawe-Taylor et al., 2004; Scholkopf & Smola, 2018). While these methods are highly expressive by computing with an infinite-dimensional feature space using the "kernel trick," most methods require solving linear systems with the kernel matrix—an operation that scales cubically with the number of data points. This is prohibitively expensive for increasingly large modern datasets and fundamentally limits the applicability of kernel methods.

To remedy this, a large number of methods have been developed that accelerate operations involving kernel matrices. Typically, these methods provide faster matrix vector products and may be paired with classical iterative methods to solve the necessary linear systems. For example, in the machine learning community, a popular approach is the Nyström method, which constructs a low-rank approximation based on a random sample of a kernel matrix's columns (Williams & Seeger, 2001; Drineas et al., 2005; Kumar et al., 2009, 2012). In the context of Gaussian process (GP) regression, Snelson & Ghahramani (2005) introduced the usage of inducing points, leading to $\mathcal{O}(Nm^2)$ runtime for N data points and m inducing points. In Section 3, we develop a new scheme for this problem based on analytical expansions which can be readily applied to a broad range of kernel functions that arise in a diverse set of applications—a feature we highlight in Section 5.

A related area of techniques are random feature methods (Rahimi & Recht, 2007; Li et al., 2019). These methods can generate low-rank kernel matrix approximations via random sampling from the spectral density based on Bochner's Theorem. Unlike these methods,

the FKT requires only derivatives of the kernel rather than its Fourier transform and does not need to consider sampling strategies. However, the FKT cannot handle higher dimensional datasets as well in its current form.

In scientific computing and applied mathematics, a large body of work concerns the acceleration of physical simulations in which the force two particles exert on each other is modeled by a kernel function, like the inverse-square law $\sim 1/\|\mathbf{x}-\mathbf{y}\|^2$ for gravitational and electromagnetic forces. Famously, Greengard & Rokhlin (1987) introduced the Fast Multipole Method (FMM), which provides linear-time computation of approximate matrix-vector multiplications with certain Green's function kernel matrices based on analytical expansions. The Fast Gauss Transform (FGT) (Greengard & Strain, 1991) applied similar analysis to the Gaussian kernel, and was subsequently improved to enable efficient computations in higher dimensions (Yang et al., 2003) and applied to kernel-based machine learning methods (Yang et al., 2004). Importantly, in these cases it is possible to derive concrete error bounds based on the analytical expansions. However, extending these methods relies on extensive work per kernel and is dependent on finding/developing appropriate analytical expansions. In contrast, our method leverages a new general analytical expansion to allow for immediate application to a variety of kernels. Even with this generality, we are still able to provide bounds and computational complexity analysis in Section 4 that is experimentally demonstrated in Section 5.

Contribution In this work, we propose the Fast Kernel Transform (FKT), an algorithm that allows for matrix-vector multiplication with kernel matrices in $\mathcal{O}(N \log N)$ operations and is applicable to any isotropic kernel which is analytic away from the origin and any dataset in moderate dimensions ($d < 7$). The FKT achieves this combination of computational efficiency and broad applicability by leveraging a new general analytical expansion introduced herein, which is implemented in Julia using modern computer algebra and auto-differentiation technologies and is provided open-source. We demonstrate the FKT's scaling on synthetic data and compare it with adjacent techniques in approximate kernel matrix approximation, and apply the FKT to stochastic neighborhood embedding (t-SNE) and Gaussian process regression using real-world oceanographic data to highlight the method's versatility.

2 PRIOR WORK

Algorithms that compute (approximate) matrix vector products with kernel matrices have a long history and include algorithms of various flavors. Simplistically, these methods either leverage a regular grid in the underlying domain or adaptive decompositions, and either use analytical expansions for kernel functions or purely computational schemes for compression. Concretely, our FKT leverages adaptive decompositions and a semi-analytic scheme for compressing long-range interactions.

Adaptive Methods The need for fast summation methods in N-body problems for unstructured data (i.e., matrix vector products with specific kernels) drove the development of methods that take advantage of two key features: (1) adaptive decompositions of the underlying spatial domain and (2) the ability to compress interactions between points that are well separated. This led to the development of the Barnes-Hut algorithm (Barnes & Hut, 1986) and the FMM (Greengard & Rokhlin, 1987) for computing matrix vector products. While the FMM attains $\mathcal{O}(N)$ scaling (with a constant that depends mildly on the desired accuracy), it explicitly leverages an analytical expansion for the underlying kernel and associated translation operators. Therefore, extending the algorithm to additional kernels requires extensive work. This has been done for e.g. the Helmholtz kernel $K(r) = e^{i\kappa r}/(4\pi r)$ via the use of Bessel and Hankel functions (Greengard et al., 1998).

To expand the applicability of these adaptive methods to more general kernels, numerical schemes were developed to compress long-range kernel interactions. These schemes led to algorithms such as the kernel independent FMM (Ying et al., 2004; Ying, 2006; Mikhalev & Oseledets, 2016), memory efficient kernel approximation (MEKA) (Si et al., 2017), and, more generally, so-called rank-structured factorizations and fast direct methods for matrices (see, e.g., (Martinsson, 2019; Hackbusch, 2015) for such methods in the context of integral equations). Moreover, these methods have been successfully applied to Gaussian process regression (Börm & Garcke, 2007; Ambikasaran et al., 2015; Minden et al., 2017; Chen et al., 2017). While broadly applicable, these methods can be sub-optimal if analytical expansions for kernel functions are available, as they rely on algebraic factorizations such as the interpolative decomposition (Cheng et al., 2005).

Grid-Based Methods For certain data distributions it can be advantageous to leverage regular grids on the computational domain to accelerate matrix vector products (and/or build effective pre-conditioners). Notably, if the observation points lie on a regular grid and the kernel function has certain structural properties it is possible to leverage the Fast Fourier Transform (FFT) to compute matrix vector products in $\mathcal{O}(N \log N)$ time.

However, observation points typically do not lie precisely on a regular grid. The so-called pre-corrected FFT (Phillips & White, 1994; White et al., 1994) solves this problem by incorporating aggregation and interpolation operators to allow for computations using a regular grid that are then accelerated by the FFT. An analogous method called structured kernel interpolation (SKI) is popular within the Gaussian process community (Wilson & Nickisch, 2015) as an acceleration of the so-called inducing point method (Snelson & Ghahramani, 2005).

3 THE FAST KERNEL TRANSFORM

We are interested in computing the matrix-vector product

$$z_i = \sum_{j=0}^{N} K(\|\mathbf{r}_i - \mathbf{r}_j\|) y_j. \tag{1}$$

where y is a given vector of real or complex numbers, $\mathbf{r}_i \in \mathbb{R}^d$ for $i = 0, \ldots, N$, and K is an isotropic kernel. Henceforth, we will overload notation to say that $K_{ij} \coloneqq K(\|\mathbf{r}_i - \mathbf{r}_j\|)$ and (1) can be written as $z = Ky$. The technique we propose is based on the famous Barnes-Hut (Barnes & Hut, 1986) style of tree-code algorithm. A tree decomposition is performed of the space containing the dataset's points, and for each tree node, we compute a set of distant points whose kernel interactions with the node's points can be compressed. In the original Barnes-Hut scheme, this compression is done by summing interactions with the center of mass—in our scheme we generalize this to a new multipole expansion which can more accurately represent the points inside the node. Compressing these interactions will produce low-rank approximations for large off-diagonal blocks of the kernel matrix, yielding an efficient matrix multiplication algorithm. We review each of these components in the following sections.

3.1 TREE DECOMPOSITION

We use a decomposition inspired by the binary partitioning of the k-d tree (Bentley, 1975). This scheme begins with a single hypercube root node containing all points, and iteratively splits nodes into pairs of child nodes via axis-aligned separating hyperplanes. At each split, the hyperplane is chosen to (a) split the node in half, (b) keep the aspect ratio (the maximum ratio between pairs of node side lengths) below two, and (c) optimally divide the points evenly while satisfying the first two constraints. These qualities are chosen to encourage hyperrectangular nodes with minimal aspect ratio. Low aspect ratio is desired as it will correspond to smaller sets of nearby points and hence fewer dense computations. When a node contains fewer than some prescribed threshold of points, it is not split and becomes a leaf node which has no children. An example of this decomposition is shown in Figure 1.

Once a domain decomposition is computed, our algorithm requires, for every tree node i, a set F_i of far points which are far enough from the node to allow accurate compression. To be precise, F_i is defined as

$$F_i \coloneqq \left\{ \mathbf{r} \mid \max_{\mathbf{r}' \in \text{node}} \frac{\|\mathbf{r}' - \mathbf{r}_c\|}{\|\mathbf{r} - \mathbf{r}_c\|} < \theta \right\} \setminus \bigcup_{j \in \mathcal{A}(i)} F_j, \tag{2}$$

where $\mathbf{r}_c$ is the centroid of the relevant node's points, $\mathcal{A}(i)$ refers to all ancestors of node i, and θ is an input distance parameter. The two components of the right side of (2) are meant to ensure that we only compress distant enough interactions, and we only compress interactions once. The distance parameter θ may then be varied to trade-off accuracy and computation time.

3.2 FAST MATRIX-VECTOR MULTIPLICATION

Once the sets of far points are generated for all nodes, the FKT proceeds as described in Algorithm 1. For each node i, we use a low-rank approximation of the kernel to compute interactions between points in the node and those in the F_i. Furthermore, for each leaf l we use dense computations for interactions between points in the leaf and its nearby points N_l, where N_l is defined as

$$N_l \coloneqq \left\{ \mathbf{r} \mid \max_{\mathbf{r}' \in \text{leaf}} \frac{\|\mathbf{r}' - \mathbf{r}_c\|}{\|\mathbf{r} - \mathbf{r}_c\|} \geq \theta \right\}, \tag{3}$$

where $\mathbf{r}_c$ is the centroid of the relevant leaf's points.

In summary the approximation is given by

$$\begin{aligned} z = Ky &= \sum_{l \in \text{leaves}} K_{N_l, l} * y_l + \sum_{b \in \text{nodes}} K_{F_b, b} * y_b \\ &\approx \sum_{l \in \text{leaves}} K_{N_l, l} * y_l + \sum_{b \in \text{nodes}} \widehat{K}_{F_b, b} * y_b, \end{aligned}$$

where $K_{N_l,l}$ is the submatrix of K whose columns correspond to points in the leaf node l and whose rows correspond to points in the near field N_l of the leaf node l, $K_{F_b,b}$ is the analogous submatrix for any node b and its far field F_b, y_l and y_b are the subvectors of y corresponding to the points in the leaf l or node b respectively, and $\widehat{K}_{F_b,b}$ is a low rank approximation to the typically large $K_{F_b,b}$. In Algorithm 1, $s2m$ and $m2t$ refer to "source-to-multipole" and "multipole-to-target" matrices respectively, and collectively form the low-rank approximation $\widehat{K}_{F_b,b}$. Their entries will be described in Section 3.4.

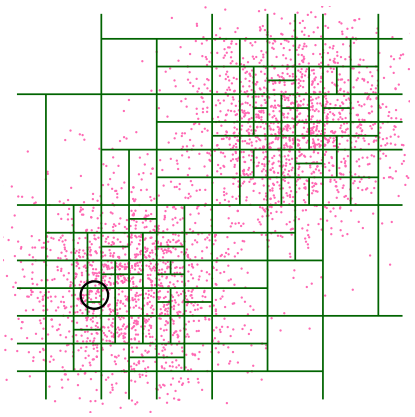

Figure 1: 2D domain decomposition on points from a Gaussian mixture. Points outside the circle are considered distant enough for compression with the circled box, for a certain θ in (2).

Algorithm 1 Barnes-Hut with Multipoles

```
tree ← BinarySpacePartitioning(points)
z ← 0
for n ∈ tree.nodes do
  {Compute compressed far interactions.}
  s2m ← source2mult(n)
  m2t ← mult2target(n)
  z[n.far] += m2t ∗ (s2m ∗ y[n.indices])
  if isleaf(n) then
    {Compute nearby dense interactions.}
    near_mat ← K(n.near, n.indices)
    z[n.near] += near_mat ∗y[n.indices]
  end if
end for
return z
```

3.3 LOW-RANK KERNEL APPROXIMATIONS

Given the preceding approach, the key to a fast algorithm is the availability of a sufficiently accurate low-rank approximation $\widehat{K}_{F_b,b} \approx K_{F_b,b}$ valid when the sets F_b and b contain well-separated points. Our approach to building these approximations is inspired by multipole methods (specifically the FMM (Greengard & Rokhlin, 1987)) for solving the N-body problem (1) when K is the electrostatic potential. If $|b| = M$ and $|F_b| = N$, multiplying by this matrix requires $\mathcal{O}(MN)$ work. However, if we have access to a low rank approximation

$$K(\|\mathbf{r}_i - \mathbf{r}_j\|) \approx \sum_{k=0}^{\mathcal{P}} U_k(\mathbf{r}_i) V_k(\mathbf{r}_j) \tag{4}$$

valid for $i \in b$ and $j \in F_b$ we can use it to accelerate the computation. Specifically, using (4) we can rewrite (1) as

$$z_i \approx \sum_{j=0}^{N} \sum_{k=0}^{\mathcal{P}} U_k(\mathbf{r}_i) V_k(\mathbf{r}_j) y_j = \sum_{k=0}^{\mathcal{P}} U_k(\mathbf{r}_i) \sum_{j=0}^{N} V_k(\mathbf{r}_j) y_j$$

and the two sums may be computed in $\mathcal{O}(\mathcal{P}(M+N))$ time. In this case, the V_k sum corresponds to the $s2m$ matrix in Algorithm 1 and the U_k sum corresponds to the $m2t$ matrix.

For example, let $\mathbf{r}', \mathbf{r} \in \mathbb{R}^3$. A classic example of an expansion of the form in (4) which is low rank for well-separated points is the multipole expansion of the electrostatic potential

$$K(\|\mathbf{r}'-\mathbf{r}\|) = \frac{1}{\|\mathbf{r}' - \mathbf{r}\|} = \frac{1}{\|\mathbf{r}\| \sqrt{1 + \frac{\|\mathbf{r}'\|}{\|\mathbf{r}\|}\left(\frac{\|\mathbf{r}'\|}{\|\mathbf{r}\|} - 2\cos\gamma\right)}}$$

where γ is the angle between $\mathbf{r}'$ and $\mathbf{r}$. Expanding in powers of $\frac{\|\mathbf{r}'\|}{\|\mathbf{r}\|}$ yields the expansion in Legendre polynomials

$$K(\|\mathbf{r}' - \mathbf{r}\|) = \frac{1}{\|\mathbf{r}\|} \sum_{k=0}^{\infty} \left(\frac{\|\mathbf{r}'\|}{\|\mathbf{r}\|}\right)^k P_k(\cos\gamma). \tag{5}$$

This may be put into the form of (4) by splitting $P_k(\cos\gamma)$ into functions of $\mathbf{r}'$ and $\mathbf{r}$ using the spherical harmonic addition theorem (see Sec. 12.8 in (Arfken, 1985)).

$$\frac{2k+1}{4\pi} P_k(\cos\gamma) = \sum_{h=-k}^{k} Y_k^h(\mathbf{r}') Y_k^h(\mathbf{r})^*, \tag{6}$$

where Y_k^h are spherical harmonics. The FKT leverages modern computational tools to build analogous low-rank approximations for a broad class of kernels.

3.4 THE GENERALIZED MULTIPOLE EXPANSION

We build our new technique by developing an expansion for general kernels into separable radial and angular functions as in (5). We begin by defining $\varepsilon := \frac{\|\mathbf{r}'\|}{\|\mathbf{r}\|}\left(\frac{\|\mathbf{r}'\|}{\|\mathbf{r}\|} - 2\cos\gamma\right)$, where γ is again the angle between $\mathbf{r}'$ and $\mathbf{r}$. Then $K(\|\mathbf{r}' - \mathbf{r}\|) = K(\|\mathbf{r}\|\sqrt{1+\varepsilon})$ by the law of cosines, and, assuming $\|\mathbf{r}\| > 0$ and K is analytic except possibly at the origin, we can form a Taylor expansion around $\varepsilon = 0$

$$K(\|\mathbf{r}' - \mathbf{r}\|) = \sum_{n=0}^{\infty} \frac{\varepsilon^n}{n!} \frac{\partial^n}{\partial \varepsilon^n} K(\|\mathbf{r}\|\sqrt{1+\varepsilon})_{\varepsilon=0}. \tag{7}$$

By expanding the ε^n terms via the binomial theorem, transforming from powers of cosine into Gegenbauer polynomials of cosine (via an identity from Avery

(1989)), and using Faa di Bruno's theorem for the derivatives with respect to ε, this sum can be rewritten as an expansion in (hyper)spherical harmonics, as given by Theorem A.3

Theorem 3.1. *If K is analytic except possibly at the origin, then for $\mathbf{r}'$,$\mathbf{r}$ within the radius of convergence,*

$$K(\|\mathbf{r}'-\mathbf{r}\|)=\sum_{k=0}^{\infty}\sum_{h\in\mathcal{H}_k}Y_k^h(\mathbf{r})Y_k^h(\mathbf{r}')^*\mathcal{K}^{(k)}(\|\mathbf{r}'\|,\|\mathbf{r}\|)$$

where Y_k^h are hyperspherical harmonics (see Avery & Avery (2018)),

$$\mathcal{H}_k\coloneqq\{(\mu_1,\dots\mu_{d-2}):k\geq\mu_1\geq\cdots\geq|\mu_{d-2}|\geq 0\},$$

$$\mathcal{K}^{(k)}(\|\mathbf{r}'\|,\|\mathbf{r}\|)\coloneqq\sum_{j=k}^{\infty}\|\mathbf{r}'\|^j\sum_{m=1}^{j}K^{(m)}(\|\mathbf{r}\|)\|\mathbf{r}\|^{m-j}\mathcal{T}_{jkm}^{(\alpha)},\tag{8}$$

and $\mathcal{T}_{jkm}^{(\alpha)}$ are constants which depend only on the dimension and not on the kernel or data. The radius of convergence is the same as that of (7).

(See Supplemental Material for the proof and the definition of $\mathcal{T}_{jkm}^{(\alpha)}$). We thus arrive at the approximation underlying the Fast Kernel transform, a truncated expansion with truncation parameter p of the form

$$K(\|\mathbf{r}'-\mathbf{r}\|)\approx\sum_{k=0}^{p}\sum_{h\in\mathcal{H}_k}Y_k^h(\mathbf{r})Y_k^h(\mathbf{r}')^*\mathcal{K}_p^{(k)}(\|\mathbf{r}'\|,\|\mathbf{r}\|),\tag{9}$$

where $\mathcal{K}_p^{(k)}$ is the truncation of the infinite sum in (8) made by replacing ∞ with p. To see the generalized multipole expansion (9) is of the form in (4), we introduce multi-indices (as in the FMM)

$$K(\|\mathbf{r}'-\mathbf{r}\|)=\sum_{k=0}^{p}\sum_{h\in\mathcal{H}_k}\sum_{j=k}^{p}U_{k,h,j}(\mathbf{r}')V_{k,h,j}(\mathbf{r}),$$

where

$$U_{k,h,j}(\mathbf{r}')\coloneqq Y_k^h(\mathbf{r}')^*\|\mathbf{r}'\|^j,$$
$$V_{k,h,j}(\mathbf{r})\coloneqq Y_k^h(\mathbf{r})\sum_{m=1}^{j}K^{(m)}(\|\mathbf{r}\|)\|\mathbf{r}\|^{m-j}\mathcal{T}_{jkm}.$$

An FKT implementation will generate the $s2m$ and $m2t$ matrices in Algorithm 1 by collecting the $U_{k,h,j}(\mathbf{r}')$ functions as columns into a tall and skinny $s2m$ matrix and the $V_{k,h,j}(\mathbf{r})$ functions as rows into a short and fat $m2t$ matrix for each node in the tree. The latter requires evaluation of derivatives of the kernel, which may be done via automatic differentiation.

The sums over j and k in the definition of $\mathcal{K}^{(k)}$ turns out to have interesting and helpful properties for our algorithm. In particular, for certain types of kernels it is possible to automatically compute more concise expansions than the form given in (8), resulting in better complexity. The details of this additional compression can be found in the Supplemental Material.

4 ANALYSIS

4.1 TRUNCATION ERROR

Lemma 4.1 (Truncation Error)**.** *The truncation* (9) *yields error $|\mathcal{E}_P|$ where*

$$|\mathcal{E}_P|\leq\sum_{k=0}^{\infty}\binom{k+d-3}{k}\times\left|\sum_{j=\max(p+1,k)}^{\infty}\sum_{m=1}^{j}K^{(m)}(\|\mathbf{r}\|)\|\mathbf{r}\|^m\left(\frac{\|\mathbf{r}'\|}{\|\mathbf{r}\|}\right)^j\mathcal{T}_{jkm}^{(\alpha)}\right|.\tag{10}$$

Proof. This follows from the bound $|C_k^{(\alpha)}(\cos\gamma)|\leq\binom{k+d-3}{k}$ on Gegenbauer polynomials (DLMF, Eq. 18.14.4) and bringing the absolute value inside the sum. □

In Figure 3, we report several empirical findings on this bound. As in the error analysis of the FMM for the electrostatic and Helmholtz kernels, the error is observed to decay exponentially with the choice of truncation parameter. In practice, the above bound turns out to be fairly loose—as we report in Section 5, a choice of $p=4$ yields a residual less than 10^{-4} for reasonable distance criteria. Because the bound in Lemma 4.1 is observed to be loose (albeit descriptive) in practice, we omit further analysis. It is of interest to further develop and analyze tighter upper bounds.

4.2 COMPUTATIONAL COMPLEXITY

To assess the computational complexity of the FKT, we need to understand the size of our compressed far-field expansion. Our low rank approximation takes the form

$$K(\|\mathbf{r}'-\mathbf{r}\|)\approx\sum_{k=0}^{\mathcal{P}}U_k(\mathbf{r}')V_k(\mathbf{r})$$
$$=\sum_{k=0}^{p}\sum_{h\in\mathcal{H}_k}Y_k^h(\mathbf{r})Y_k^h(\mathbf{r}')^*\mathcal{K}_p^{(k)}(\|\mathbf{r}'\|,\|\mathbf{r}\|),$$

and it is not hard to show (see Supplemental Material) that $\mathcal{P}=\binom{p+d}{d}\sim d^p$. We note that this is the same as the number of terms in the expansion underlying the Improved FGT (Yang et al., 2003), and is achieved for a much broader class of kernels by the FKT.

Table 1: Commonly used covariance functions in Gaussian process regression

Exponential	$K(r) = e^{-r}$
Matérn ($\nu = 3/2$)	$\sigma^2(1 + \frac{\sqrt{3}r}{\rho})e^{-\sqrt{3}r/\rho}$
Cauchy	$\frac{1}{1+r^2/\sigma^2}$
Rational Quadratic ($\alpha = 1/2$)	$\frac{1}{\sqrt{1+r^2/\sigma^2}}$

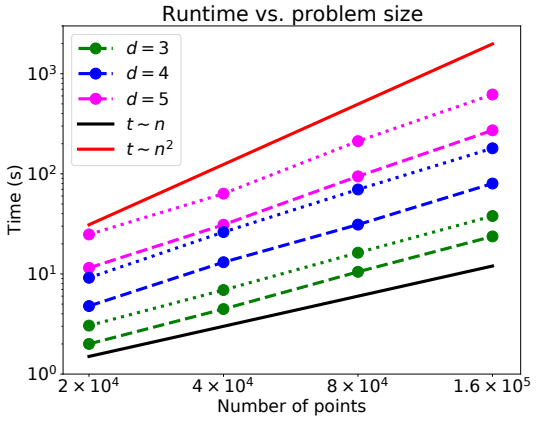

Figure 2: Runtimes of the FKT for matrix-vector multiplies for the Matérn kernel with $\nu = 1/2$. Dashed lines show results for $p = 4$ and dotted lines show $p = 6$.

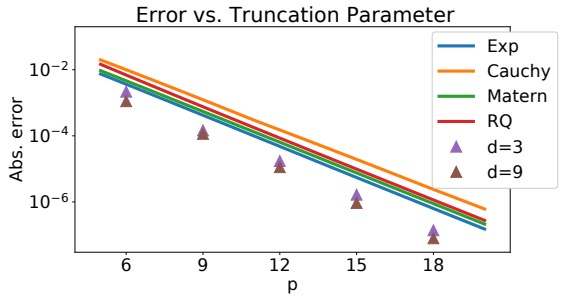

Figure 3: The lines are estimates of the upper bound for $d = 3$ in (10) for various kernels found by fixing $\mathbf{r}'/\mathbf{r} = 1/2$, summing from $p + 1$ to 30, and taking the maximum over 2000 uniformly chosen points in $\|\mathbf{r}\| \in [0, 20]$ (we do not see growth of error with $\|\mathbf{r}\|$). These estimates are shown for the Exponential, Matérn, Cauchy, and Rational Quadratic kernels as described in Table 1. The triangles are experimentally observed errors which are calculated for the Cauchy kernel FKT approximation by taking the maximum absolute error of the truncated expansion for 1000 randomly selected pairs of points $\mathbf{r}', \mathbf{r}$ satisfying $|\mathbf{r}'| = 1, |\mathbf{r}| = 2$.

The complexity of Algorithm 1 is the sum of the cost of computing the dense matrices for nearby interactions, the cost of computing the $s2m$ matrices for every node, and the cost of computing the $m2t$ matrices for every node. For simplicity of this analysis, we assume that every leaf has at most m points, each leaf has at most N_d points in its near field, and each point is in the far field of at most F_d nodes. If the total number of points is N, the total cost is given by

$$\begin{aligned}\text{FKT}_{cost} &= \mathcal{O}\left(\frac{N}{m}mN_d + N\log\left(N/m\right)\mathcal{P} + NF_d\mathcal{P}\right)\\ &= \mathcal{O}\left(N\left(N_d + \log\left(N/m\right)d^p + F_d d^p\right)\right).\end{aligned}$$

In practice, F_d is generally exponential in the dimension and has an additional factor of $\log(N/m)$ coming from the depth of the tree. N_d depends on the maximum leaf capacity m and a factor exponential in the dimension. Letting d be the dimension, we have

$$\text{FKT}_{cost} = \mathcal{O}\left(N\left(mc_n^d + (1+c_f^d)\log(N/m)d^p\right)\right) \quad (11)$$

where c_n, c_f are constants which depend on the problem geometry, typically between 2 and 5. In the worst case we have $mc_n^d \sim N$ (for example, when the distinction between near and far sets diminishes in higher dimensions); then the dense nearby interactions will overwhelm the compressed far interactions and the complexity will be $\mathcal{O}(N^2)$. However, when there are sufficient sets of well-separated points (more common for $d < 6$), the computations for far points dominate and we have

$$\text{FKT}_{cost} = \mathcal{O}\left(N\log(N/d^p) \times c_f^d \times d^p\right) \quad (12)$$

where we have set $m = \mathcal{O}(d^p)$. In cases where the additional compression described at the end of Section 3.4 is applied, the size $\mathcal{P}$ of the expansion can be reduced by a factor of d and the d^p term in (12) becomes d^{p-1}.

4.3 LIMITATIONS

The FKT will currently not scale well to dimensions greater than 6, although its underlying expansions remain accurate. The problem is that the method requires dense computation of points nearby each other, which leads to poor scaling in high dimensions when points tend to be closer together. In contrast, the FGT provide a low-rank approximation for points nearby to each other based on the global low-rankness of the Gaussian kernel. Although the FKT can provide low-rank approximations for distant points, it cannot yet do so for nearby points if they exist.

Although the FKT automatically finds the analytical expansions foundational to the FMM, it scales quasi-linearly rather than linearly as the FMM does. One

way to make the FKT a linear algorithm would be to develop FMM-style translation operators for the expansion general to any kernel.

Finally, although we present a theoretical bound on the error of the expansion for a pair of points, it does not give a clear intuition for the behavior of the error with p, or the end-to-end error in a MVM accelerated by compression via the expansion. To supplement this absence of intuitive theoretical guarantees, we present empirical studies of the behavior of the error with respect to relevant parameters for several kernels, but future developments should provide deeper illumination into the error guarantees that can be given for kernels with known bounds on their derivatives.

5 EXPERIMENTS

We have implemented the FKT in Julia as part of an open source toolkit[1], making use of the NearestNeighbors.jl package (Carlsson et al., 2020) to compute near and far sets of points, and the TaylorSeries.jl package (Benet & Sanders, 2019) to automatically compute derivatives. Both packages are licensed under the MIT "Expat" License. The synthetic experiments were performed single-threaded on a 2020 Apple Macbook Air with an M1 CPU and 8GB of RAM, and the regression experiment was performed on a 2017 MacBook with a Dual-Core Intel Core i7 and 16GB of RAM.

5.1 SYNTHETIC DATA

To test the runtime of the algorithm, we generate a synthetic dataset of points uniformly distributed on a unit hypersphere. We then approximate a matrix-vector multiplication with a Matérn kernel matrix (see Table 1) on this dataset against a random vector. Our test uses a distance parameter value of $\theta = 0.75$, maximum leaf capacity of 512, and includes results for truncation parameter $p = 4, 6$. Results for this test in a variety of dimensions and problem sizes are shown in Figure 2—the runtime is seen to be quasi-linear in the problem size. Moreover, the FKT becomes faster than dense matrix multiplication at $N = 1000$ for $d = 3$, $N = 5000$ for $d = 4$, and $N = 20,000$ for $d = 5$. To test the accuracy of the approximation, we compare the truncated expansion to the true kernel value for the Cauchy kernel in 3 and 9 dimensions. The errors are calculated for the p-term approximation for 1000 randomly selected pairs of points $\mathbf{r}', \mathbf{r}$ satisfying $|\mathbf{r}'| = 1, |\mathbf{r}| = 2$, and p is swept from 6 to 18 (see Figure 3). The error is seen to decay exponentially with p, and not be affected by dimension. Results for this experiment in more dimensions and for more kernels can be found in the Supplemental Material.

To test how the FKT performs in comparison to other approximate kernel MVM techniques, we ran a set of experiments using the FKT, SKI, subset of regressors (SoR) method (Silverman, 1985), fully independent training conditional (FITC) method (Snelson & Ghahramani, 2006), and FGT (when applicable). For SKI, SoR, and FITC we use GPyTorch implementations (Gardner et al., 2018a), and for the FGT we use the C++ figtree implementation in Morariu et al. (2008). All techniques are applied to the problem of multiplying a random vector by a kernel matrix on a set of 2^{14} points uniformly distributed on the surface of a sphere in $\mathbb{R}^3$. Figure 4 shows accuracy/cost tradeoff plots for three different kernels—the FKT is seen to have strong comparative performance in the high-accuracy regime for non-smooth kernels. SKI suffers from poor interpolation of the non-smooth kernels, and the SoR and FITC methods do not efficiently capture the full-rank near-diagonal information within their respective low-rank and diagonal-plus-low-rank structures.

5.2 STOCHASTIC NEIGHBORHOOD EMBEDDING

The stochastic neighborhood embedding (SNE) was proposed by Hinton & Roweis (2002), and Van Der Maaten & Hinton (2008) followed-up that work with the improved t-distributed SNE (t-SNE). The t-SNE is widely used as an effective tool for dimensionality reduction for data visualization. An implementation of its optimization routine requires sums of and matrix-vector-products with kernel matrices with N^2 entries. In particular, the relevant gradient of the t-SNE objective contains matrix-vector products with a kernel matrix of the Cauchy kernel $(1+\|\mathbf{r}\|^2)^{-1}$ with two-dimensional inputs, which is a prime candidate for the application of FKT. Previously, Van Der Maaten (2014) proposed accelerated methods for t-SNE based on tree codes including the aforementioned Barnes-Hut scheme. While the Barnes-Hut scheme is simpler, Figure 5, left, shows that FKT exhibits a superior accuracy-runtime tradeoff if more accuracy is desired. The plot was generated by varying the θ distance parameter as in Van Der Maaten (2014). While very high accuracy might not be of utmost concern for optimization in t-SNE, which has a more qualitative goal, this result more generally shows that FKT achieves a high degree generality, since it needs *no manual adaption* to work on the relevant matrices and compute the visualization of MNIST (LeCun & Cortes, 2010) in Figure 5, right. The MNIST data is licensed under the CC BY-SA 3.0 license.

[1] https://github.com/jpryan1/FastKernelTransform.jl

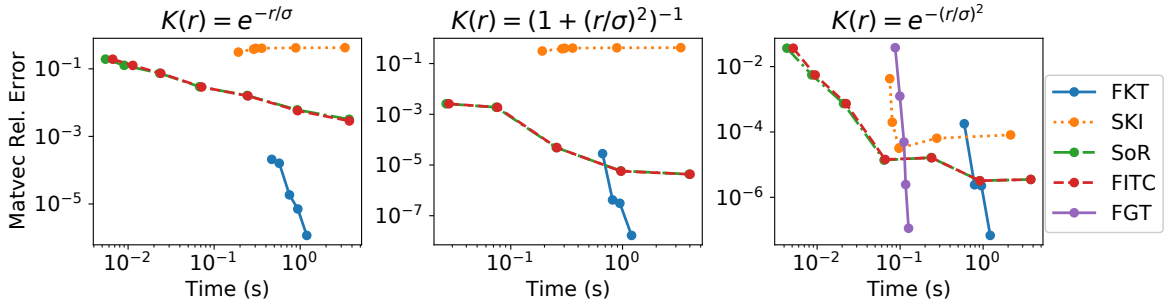

Figure 4: Accuracy/cost tradeoff for several approximate kernel matvec techniques applied to four different kernel problems. The dataset consists of 2^{14} points uniformly distributed on the surface of a sphere in $\mathbb{R}^3$, and the lengthscale parameter σ is set to 0.25 for all kernels. To generate the tradeoff curves, we varied the truncation parameter p in the FKT, number of inducing points (random subset of original dataset) for SoR and FITC, grid size for SKI, and error tolerance for FGT. As is clearly illustrated, the FKT has a favorable accuracy/cost tradeoff and is particularly favorable when higher accuracy is desired.

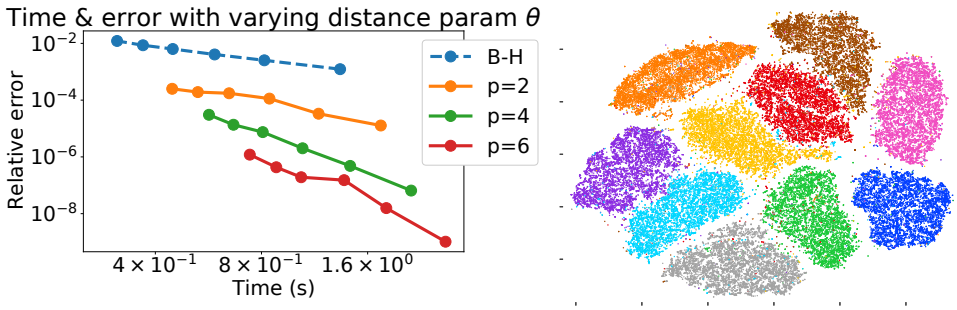

Figure 5: Left: Runtimes and relative errors for a series of matrix vector multiplies using the Cauchy kernel on a 2D dataset of 20k uniformly distributed points in the unit square. B-H refers to the Barnes-Hut method, which is equivalent to the $p = 0$ FKT with centers of masses as the expansion centers. The maximum leaf capacity was 512, and for each p, we varied the distance parameter θ between 0.25 and 0.75. Right: A t-SNE embedding of the MNIST training set of 60,000 images of digits computed via application of FKT.

5.3 GAUSSIAN PROCESS REGRESSION OF SEA SURFACE TEMPERATURE

Gaussian processes (GPs) constitute another important class of a kernel methods. Importantly, inference of the posterior predictive mean of a GP can be carried out exclusively through matrix-vector multiplications with kernel matrices and a diagonal "noise" variance matrix (Wang et al., 2019). To highlight the generality of FKT, we use it here to compute a GP regressor on sea surface temperature data from Copernicus, the European Union's Earth Observation Programme (Merchant et al., 2019), which is licensed under the CC-BY 4.0 license. The dataset is collected by a satellite orbiting the earth several times per day, leading to measurement locations with a complex spatial structure (see Figure 6, left). Each data point comes with an uncertainty estimate, which we use to populate the diagonal noise variance matrix of the model. We consider data for the first seven days of 2019, for which more than 8 million data points were collected and sub-sampled it down to a still considerable 145,913 observations by taking every 56th data point in the temporal order in which they were collected. We then evaluated the posterior predictive mean of a GP with the Matérn-3/2 kernel conditioned on the observations and their uncertainties at 480,000 predictive points to arrive at the result on the right of Figure 6. We restricted the predictions to be within 60 degrees of latitude of the equator, since the satellite data is very sparse in the polar regions. Table B.2 in the Supplemental Material provides error statistics showing that the FKT has virtually the same extrapolation performance as an exact method on the test data, in contrast to SKI and FITC which must sacrifice predictive accuracy when constrained to have a comparable runtime. The entire computation completed in around twelve minutes on a 2017

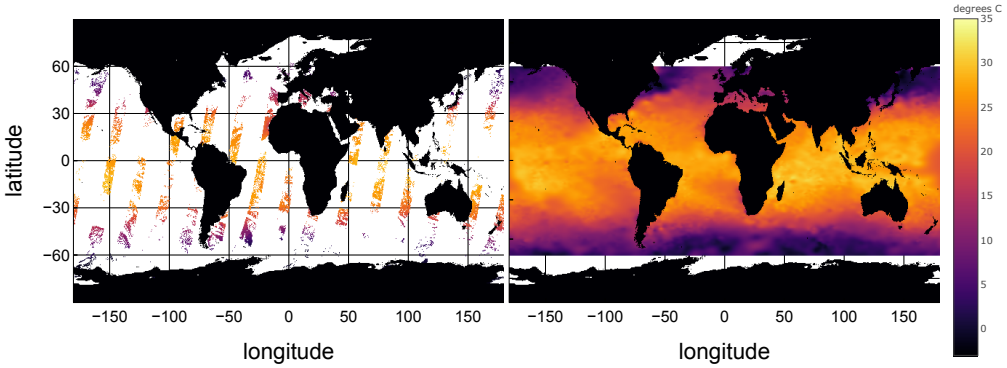

Figure 6: Sea surface temperatures collected by a single satellite throughout one day (left) and the posterior mean of a Gaussian process with a Matérn kernel conditioned on seven days of data.

MacBook with a Dual-Core Intel Core i7 and 16GB of RAM, highlighting once more the rare combination of generality and high efficiency that FKT achieves.

6 DISCUSSION AND BROADER IMPACTS

We have presented the Fast Kernel Transform, a general method for the automatic computation of analytical expansions of isotropic kernels which can be used in hierarchical matrix algorithms on datasets in moderate dimensions ($d < 7$). The FKT has a high, quantifiable, and controllable level of accuracy, and its cost grows only quasi-linearly in the number of data points and polynomially in the ambient dimension. While our work is entirely algorithmic in nature, it is important to remark that using approximation schemes such as the FKT can introduce additional variation in downstream tasks that are not anticipated; it is important to assess the level of sensitivity of different applications to such perturbations and validate models developed with these methods across a broad range of criteria.

The method is based on a new analytic approximation scheme whose number of terms is equal to those of the expansions developed for the Improved FGT, but for a much broader set of kernels. At its core, our method reflects a generalization of the mathematical tools underlying seminal works in kernel methods, such as the FMM and the FGT, and opens up many opportunities for further theoretical study and algorithmic development, such as work on a more rigorous foundation for the class of kernels for which the FKT excels, and work on removing the ambient dimension from the cost of FKT via an appropriate selection of harmonics to retain when an underlying intrinsically lower-dimensional manifold is known or may be discovered. Further, the logarithmic term could be removed by the creation of translation operators so as to completely generalize the FMM to this broad class of kernels. These translation operators are the subject of current development by the authors. We believe that the methods contained herein could prove useful for a wide range of practitioners and researchers of kernel methods, enabling them to apply their methods to much larger problem instances than without acceleration, and have made an open-source implementation of FKT available.

7 ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers for their thoughtful and detailed feedback and suggestions.

References

Ambikasaran, S., Foreman-Mackey, D., Greengard, L., Hogg, D. W., and O'Neil, M. Fast direct methods for gaussian processes. *IEEE transactions on pattern analysis and machine intelligence*, 38(2):252–265, 2015.

Arfken, G. *Mathematical Methods for Physicists*. Academic Press, Inc., San Diego, third edition, 1985.

Askey, R. and Ismail, M. E.-H. A generalization of ultraspherical polynomials. In *Studies in pure mathematics*, pp. 55–78. Springer, 1983.

Avery, J. *Gegenbauer Polynomials*, pp. 25–46. Springer Netherlands, Dordrecht, 1989. ISBN 978-94-009-2323-2. doi: 10.1007/978-94-009-2323-2_3. URL https://doi.org/10.1007/978-94-009-2323-2_3.

Avery, J. E. and Avery, J. S. *Hyperspherical Harmonics and Their Physical Applications*. WORLD SCIENTIFIC, 2018. doi: 10.1142/10690. URL https://www.worldscientific.com/doi/abs/10.1142/10690.

Barnes, J. and Hut, P. A hierarchical o (n log n) force-

calculation algorithm. *nature*, 324(6096):446–449, 1986.

Benet, L. and Sanders, D. P. Taylorseries.jl: Taylor expansions in one and several variables in julia. *Journal of Open Source Software*, 4(36):1043, 2019. doi: 10.21105/joss.01043. URL https://doi.org/10.21105/joss.01043.

Bentley, J. L. Multidimensional binary search trees used for associative searching. *Commun. ACM*, 18 (9):509–517, September 1975. ISSN 0001-0782. doi: 10.1145/361002.361007. URL https://doi.org/10.1145/361002.361007.

Börm, S. and Garcke, J. Approximating gaussian processes with $\mathcal{H}^2$-matrices. In *European Conference on Machine Learning*, pp. 42–53. Springer, 2007.

Businger, P. and Golub, G. H. Linear least squares solutions by householder transformations. *Numerische Mathematik*, 7(3):269–276, 1965.

Carlsson, K., Karrasch, D., Bauer, N., Kelman, T., Schmerling, E., Hoffimann, J., Visser, M., San-Jose, P., Christie, J., Ferris, A., Anthony Blaom, P., Foster, C., Saba, E., Goretkin, G., Orson, I., Samuel, O., Choudhury, S., and Nagy, T. Kristofferc/nearestneighbors.jl: v0.4.8. December 2020. doi: 10.5281/zenodo.4301693. URL https://doi.org/10.5281/zenodo.4301693.

Chan, T. F. Rank revealing qr factorizations. *Linear algebra and its applications*, 88:67–82, 1987.

Chen, J., Avron, H., and Sindhwani, V. Hierarchically compositional kernels for scalable nonparametric learning. *Journal of Machine Learning Research*, 18(66):1–42, 2017. URL http://jmlr.org/papers/v18/15-376.html.

Cheng, H., Gimbutas, Z., Martinsson, P. G., and Rokhlin, V. On the compression of low rank matrices. *SIAM J. Sci. Comput.*, 26(4):1389–1404, April 2005. ISSN 1064-8275. doi: 10.1137/030602678. URL https://doi.org/10.1137/030602678.

Cheng, H., Crutchfield, W., Gimbutas, Z., Greengard, L., Ethridge, J., Huang, J., Rokhlin, V., Yarvin, N., and Zhao, J. A wideband fast multipole method for the helmholtz equation in three dimensions. *Journal of Computational Physics*, 216(1):300–325, July 2006. ISSN 0021-9991. doi: 10.1016/j.jcp.2005.12.001. Funding Information: The authors were supported in part by DARPA/AFOSR under the contracts F49620-03-C-0052 and F49620-03-C-0041, and by DARPA under contract HR0011-05-P-0001.

De G. Matthews, A. G., Van Der Wilk, M., Nickson, T., Fujii, K., Boukouvalas, A., León-Villagrá, P., Ghahramani, Z., and Hensman, J. Gpflow: A gaussian process library using tensorflow. *The Journal of Machine Learning Research*, 18(1):1299–1304, 2017.

Deisenroth, M. and Ng, J. W. Distributed gaussian processes. In *International Conference on Machine Learning*, pp. 1481–1490. PMLR, 2015.

DLMF. *NIST Digital Library of Mathematical Functions*. http://dlmf.nist.gov/, Release 1.1.4 of 2022-01-15. URL http://dlmf.nist.gov/. F. W. J. Olver, A. B. Olde Daalhuis, D. W. Lozier, B. I. Schneider, R. F. Boisvert, C. W. Clark, B. R. Miller, B. V. Saunders, H. S. Cohl, and M. A. McClain, eds.

Dong, K., Eriksson, D., Nickisch, H., Bindel, D., and Wilson, A. G. Scalable log determinants for gaussian process kernel learning. In *Advances in Neural Information Processing Systems*, pp. 6327–6337, 2017.

Drineas, P., Mahoney, M. W., and Cristianini, N. On the nyström method for approximating a gram matrix for improved kernel-based learning. *journal of machine learning research*, 6(12), 2005.

Gardner, J., Pleiss, G., Bindel, D., Weinberger, K., and Wilson, A. Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration. *Advances in Neural Information Processing Systems*, 2018-December:7576–7586, 2018a. ISSN 1049-5258. Funding Information: JRG and AGW are supported by NSF IIS-1563887 and by Facebook Research. GP and KQW are supported in part by the III-1618134, III-1526012, IIS-1149882, IIS-1724282, and TRIPODS-1740822 grants from the National Science Foundation. In addition, they are supported by the Bill and Melinda Gates Foundation, the Office of Naval Research, and SAP America Inc. Publisher Copyright: © 2018 Curran Associates Inc.All rights reserved.; 32nd Conference on Neural Information Processing Systems, NeurIPS 2018 ; Conference date: 02-12-2018 Through 08-12-2018.

Gardner, J., Pleiss, G., Weinberger, K. Q., Bindel, D., and Wilson, A. G. Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration. In *Advances in Neural Information Processing Systems*, pp. 7576–7586, 2018b.

Greengard, L. and Rokhlin, V. A fast algorithm for particle simulations. *Journal of computational physics*, 73(2):325–348, 1987.

Greengard, L. and Strain, J. The fast gauss transform. *SIAM Journal on Scientific and Statistical Computing*, 12(1):79–94, 1991.

Greengard, L., Huang, J., Rokhlin, V., and Wandzura, S. Accelerating fast multipole methods for the helmholtz equation at low frequencies. *IEEE Computational Science and Engineering*, 5(3):32–38, 1998. doi: 10.1109/99.714591.

Hackbusch, W. *Hierarchical matrices: algorithms and analysis*, volume 49. Springer, 2015.

Hinton, G. E. and Roweis, S. Stochastic neighbor embedding. *Advances in neural information processing systems*, 15:857–864, 2002.

Kumar, S., Mohri, M., and Talwalkar, A. Ensemble nystrom method. *Advances in Neural Information Processing Systems*, 22:1060–1068, 2009.

Kumar, S., Mohri, M., and Talwalkar, A. Sampling methods for the nyström method. *The Journal of Machine Learning Research*, 13(1):981–1006, 2012.

LeCun, Y. and Cortes, C. MNIST handwritten digit database. 2010. URL http://yann.lecun.com/exdb/mnist/.

Li, Z., Ton, J.-F., Oglic, D., and Sejdinovic, D. Towards a unified analysis of random Fourier features. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3905–3914. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/li19k.html.

Martinsson, P.-G. *Fast direct solvers for elliptic PDEs*. SIAM, 2019.

Merchant, C. J., Embury, O., Bulgin, C. E., Block, T., Corlett, G. K., Fiedler, E., Good, S. A., Mittaz, J., Rayner, N. A., Berry, D., et al. Satellite-based time-series of sea-surface temperature since 1981 for climate applications. *Scientific data*, 6(1):1–18, 2019.

Mikhalev, A. Y. and Oseledets, I. V. Iterative representing set selection for nested cross approximation. *Numerical Linear Algebra with Applications*, 23(2): 230–248, 2016. doi: https://doi.org/10.1002/nla.2021. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/nla.2021.

Minden, V., Damle, A., Ho, K. L., and Ying, L. Fast spatial gaussian process maximum likelihood estimation via skeletonization factorizations. *Multiscale Modeling & Simulation*, 15(4):1584–1611, 2017.

Morariu, V., Srinivasan, B., Raykar, V., Duraiswami, R., and Davis, L. Automatic online tuning for fast gaussian summation. volume 2008, pp. 1113–1120, 01 2008.

Phillips, J. R. and White, J. A precorrected-fft method for capacitance extraction of complicated 3-d structures. In *ICCAD*, volume 94, pp. 268–271. Citeseer, 1994.

Rahimi, A. and Recht, B. Random features for large-scale kernel machines. In Platt, J., Koller, D., Singer, Y., and Roweis, S. (eds.), *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc., 2007. URL https://proceedings.neurips.cc/paper/2007/file/013a006f03dbc5392effeb8f18fda755-Paper.pdf.

Rasmussen, C. E. and Williams, C. K. I. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005. ISBN 026218253X.

Riordan, J. Derivatives of composite functions. *Bulletin of the American Mathematical Society*, 52(8):664 – 667, 1946. doi: bams/1183509573. URL https://doi.org/.

Scholkopf, B. and Smola, A. J. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. Adaptive Computation and Machine Learning series, 2018.

Shawe-Taylor, J., Cristianini, N., et al. *Kernel methods for pattern analysis*. Cambridge university press, 2004.

Si, S., Hsieh, C.-J., and Dhillon, I. S. Memory efficient kernel approximation. *Journal of Machine Learning Research*, 18(20):1–32, 2017. URL http://jmlr.org/papers/v18/15-025.html.

Silverman, B. W. Some aspects of the spline smoothing approach to non-parametric regression curve fitting. *Journal of the Royal Statistical Society. Series B (Methodological)*, 47(1):1–52, 1985. ISSN 00359246. URL http://www.jstor.org/stable/2345542.

Snelson, E. and Ghahramani, Z. Sparse gaussian processes using pseudo-inputs. *Advances in neural information processing systems*, 18:1257–1264, 2005.

Snelson, E. and Ghahramani, Z. Sparse gaussian processes using pseudo-inputs. In Weiss, Y., Schölkopf, B., and Platt, J. (eds.), *Advances in Neural Information Processing Systems*, volume 18. MIT Press, 2006. URL https://proceedings.neurips.cc/paper/2005/file/4491777b1aa8b5b32c2e8666dbe1a495-Paper.pdf.

Van Der Maaten, L. Accelerating t-sne using tree-based algorithms. *The Journal of Machine Learning Research*, 15(1):3221–3245, 2014.

Van Der Maaten, L. and Hinton, G. Visualizing data using t-sne. *Journal of machine learning research*, 9 (Nov):2579–2605, 2008.

Wang, K., Pleiss, G., Gardner, J., Tyree, S., Weinberger, K. Q., and Wilson, A. G. Exact gaussian processes on a million data points. In *Advances in Neural Information Processing Systems*, pp. 14648–14659, 2019.

Wen, Z. and Avery, J. Some properties of hyperspherical harmonics. *Journal of Mathematical Physics*, 26 (3):396–403, 1985. doi: 10.1063/1.526621. URL https://doi.org/10.1063/1.526621.

White, J., Phillips, J., and Korsmeyer, T. Comparing precorrected-fft and fast multipole algorithms

for solving three-dimensional potential integral equations,". In *Proceedings of the Colorado Conference on Iterative Methods*, pp. 4–10. Citeseer, 1994.

Williams, C. K. and Seeger, M. Using the nyström method to speed up kernel machines. In *Advances in neural information processing systems*, pp. 682–688, 2001.

Wilson, A. and Nickisch, H. Kernel interpolation for scalable structured gaussian processes (kiss-gp). In *International Conference on Machine Learning*, pp. 1775–1784, 2015.

Yang, C., Duraiswami, R., Gumerov, N. A., and Davis, L. *Improved fast gauss transform and efficient kernel density estimation*. IEEE, 2003.

Yang, C., Duraiswami, R., and Davis, L. S. Efficient kernel machines using the improved fast gauss transform. *Advances in neural information processing systems*, 17:1561–1568, 2004.

Ying, L. A kernel independent fast multipole algorithm for radial basis functions. *Journal of Computational Physics*, 213(2):451–457, 2006.

Ying, L., Biros, G., and Zorin, D. A kernel-independent adaptive fast multipole algorithm in two and three dimensions. *Journal of Computational Physics*, 196 (2):591–626, 2004.

Supplementary Material: The Fast Kernel Transform

A TECHNICAL DETAILS

In this section, we lay out the derivation of the expansion underlying the Fast Kernel transform. Before the derivation, we review the Gegenbauer polynomials which will feature heavily. Additionally, we expand on the opportunity for additional compression for certain types of kernels alluded to in the main text. Finally we will show the details of the computation of the number of terms in the FKT expansion.

A.1 GEGENBAUER POLYNOMIALS

The generalized multipole expansion is expressed in terms of Gegenbauer polynomials, also known as ultraspherical polynomials (Askey & Ismail, 1983). For our purposes, these polynomials are best seen as generalizations of the Legendre polynomials which have higher dimensional addition theorems. They satisfy the recurrence relation

$$\begin{aligned} C_0^\alpha(x) &= 1 \\ C_1^\alpha(x) &= 2\alpha x \\ C_n^\alpha(x) &= \big[2x(n+\alpha-1)C_{n-1}^\alpha(x) - (n+2\alpha-2)C_{n-2}^\alpha(x)\big]/n \end{aligned} \tag{13}$$

and the hyperspherical harmonic addition theorem (Wen & Avery, 1985)

$$\frac{1}{Z_k^{(\alpha)}} C_k^{(\alpha)}(\cos\gamma) = \sum_{h\in\mathcal{H}_k} Y_k^h(\mathbf{r}')Y_k^h(\mathbf{r})^* \tag{14}$$

where $\mathbf{r}, \mathbf{r}' \in \mathbb{R}^d$ have angle γ between them, $\alpha = \frac{d}{2} - 1$, $Z_k^{(\alpha)}$ is a normalization term, and

$$\mathcal{H}_k \coloneqq \{(\mu_1, \dots \mu_{d-2}) : k \geq \mu_1 \geq \cdots \geq |\mu_{d-2}| \geq 0\}.$$

A.2 DERIVATION OF THE FKT EXPANSION

Throughout this section, let $r' \coloneqq \|\mathbf{r}'\|, r \coloneqq \|\mathbf{r}\|$. Before going through the proof of the main theorem of the main text, we will need a lemma concerning the application of Faa di Bruno's theorem to our particular composition of functions $(f(g(\varepsilon))$ where $g(\varepsilon) = r\sqrt{1+\varepsilon}$ and $f(g(\varepsilon)) = K(r\sqrt{1+\varepsilon})$. Before *that* lemma, we prove a combinatorial identity which will be necessary.

Lemma A.1.

$$\sum_{k=0}^{n} \binom{m+1}{k} = \sum_{k=0}^{n} \binom{2k+1}{k}\binom{m-(2k+1)}{n-k}$$

Proof. As a preliminary, note that the LHS are entries in Bernoulli's triangle, and hence satisfy

$$b_{m,n} = \begin{cases} 2^m - 1, & m = n \\ b_{m-1,n} + b_{m-1,n-1}, & m > n > 0 \\ 1, & n = 0 \end{cases}$$

It will suffice to show that the RHS follows the same recurrence. We refer to the following result from Jenson, 1902:

$$\sum_{k=0}^{n} \binom{2k+1}{k}\binom{m-(2k+1)}{n-k} = \sum_{k=0}^{n} \binom{m-k}{n-k} 2^k$$

By inspection the $m = n$ and $n = 0$ cases are immediately confirmed. If $m > n > 0$ then

$$\sum_{k=0}^{n} \binom{m-k-1}{n-k} 2^k + \sum_{k=0}^{n-1} \binom{m-k-1}{n-k-1} 2^k$$

$$= 2^n + \sum_{k=0}^{n-1} \left(\binom{m-k-1}{n-k} + \binom{m-k-1}{n-k-1} \right) 2^k$$

$$= 2^n + \sum_{k=0}^{n-1} \binom{m-k}{n-k} 2^k = \sum_{k=0}^{n} \binom{m-k}{n-k} 2^k$$

□

Lemma A.2. *When $n > 0$,*

$$\frac{\partial^n}{\partial \varepsilon^n} \left(K(r\sqrt{1+\varepsilon}) \right) |_{\varepsilon=0} = \sum_{m=1}^{n} \mathcal{B}_{nm} K^{(m)}(r) r^m \tag{15}$$

where

$$\mathcal{B}_{nm} = (-1)^{n+m} \frac{(2n-2m-1)!!}{2^n} \binom{2n-m-1}{m-1}$$

and we will use the notation $K^{(m)}(r)$ to mean $\frac{\partial^m}{\partial r^m} K(r)$

Proof. Let $g(\varepsilon) = r\sqrt{1+\varepsilon}$ and note that

$$g^{(i)}(\varepsilon)_{\varepsilon=0} = (-1)^{i+1} \frac{(2i-3)!!}{2^i} r \tag{16}$$

where we will let $(2i-3)!! = 1$ when $i = 1$. By Riordan (1946)

$$\frac{\partial^n}{\partial \varepsilon^n} \left(K(r\sqrt{1+\varepsilon}) \right) |_{\varepsilon=0} = \sum_{m=1}^{n} K^{(m)}(g(\varepsilon)_{\varepsilon=0}) \cdot B_{n,m}(g'(\varepsilon)_{\varepsilon=0}, g''(\varepsilon)_{\varepsilon=0}, ..., g^{(n-m+1)}(\varepsilon)_{\varepsilon=0})$$

where $B_{n,m}$ are the Bell polynomials (henceforth we will drop their arguments). Per usual, we set

$$B_{0,0} = 1 \quad B_{n,0} = B_{0,m} = 0$$

Then the Bell polynomials satisfy the recurrence relation

$$B_{n,m} = \sum_{i=1}^{n-m+1} \binom{n-1}{i-1} g^{(i)}(\varepsilon)_{\varepsilon=0} B_{n-i,m-1}$$

We will use this to prove

$$B_{n,m} = (-1)^{n+m} \frac{(2n-2m-1)!!}{2^n} \binom{2n-m-1}{m-1} r^m$$

$$= (-1)^{n+m} r^m \frac{(2n-m-1)!}{(m-1)!(n-m)! 2^{2n-m}} \quad n \geq m > 0$$

by induction. We begin with the base cases of $n = m = 1$ and $n > m = 1$. For the former, the recurrence relation yields

$$B_{1,1} = g'(\varepsilon)_{\varepsilon=0} = \frac{1}{2} r$$

and our claim yields

$$B_{1,1} = (-1)^2 r^1 \frac{0!}{2^1 0! 0!} = \frac{1}{2} r.$$

When $n > m = 1$ the recurrence relation gives

$$B_{n,1} = \sum_{i=1}^{n} \binom{n}{0} g^{(i)}(\varepsilon)_{\varepsilon=0} B_{n-i,0} = g^{(n)}(\varepsilon)_{\varepsilon=0} = (-1)^{(n+1)} r \frac{(2n-3)!!}{2^n}$$

and our claim yields

$$B_{n,1} = (-1)^{n+1} r^1 \frac{(2n-2)!}{2^{(2n-1)} 0! (n-1)!} = (-1)^{(n+1)} r \frac{(2n-3)!(2n-2)}{2^{(2n-2)}(n-2)!(2n-2)}$$

$$= (-1)^{(n+1)} r \frac{(2n-3)!}{2^{(2n-2)}(n-2)!} = (-1)^{(n+1)} r \frac{(2n-3)!!}{2^n}$$

For the inductive step, we need to show that

$$B_{n,m} = \sum_{i=1}^{n-m+1} \binom{n-1}{i-1} (-1)^{i+1} \frac{(2i-3)!!}{2^i} r \frac{(-1)^{n-i+m-1} r^{m-1} (2n-2i-m)!}{(m-2)!(n-i+m-1)! 2^{2n-2i-m+1}}$$

$$= r^m \frac{(-1)^{n+m}}{2^{2n-m+1}(m-2)!} \sum_{i=1}^{n-m+1} \binom{n-1}{i-1} 2^i \frac{(2i-3)!!(2n-2i-m)!}{(n-i-m+1)!}$$

Separating the $i = 1$ term out so that the double factorial is of positive integers

$$= r^m \frac{(-1)^{n+m}}{2^{2n-m+1}(m-2)!} \left(\frac{2(2n-m-2)!}{(n-m)!} + \sum_{i=2}^{n-m+1} \binom{n-1}{i-1} 2^i \frac{(2i-3)!!(2n-2i-m)!}{(n-i-m+1)!} \right)$$

Moving some terms out and rewriting the double factorial

$$= r^m \frac{(-1)^{n+m}(2n-m-1)!}{2^{2n-m}(m-1)!(n-m)!}$$

$$\cdot \left(\frac{m-1}{2n-m-1} + \frac{m-1}{(2n-m-1)!} \sum_{i=2}^{n-m+1} \binom{n-1}{i-1} 2^{i-1} \frac{(2i-3)!(2n-2i-m)!(n-m)!}{2^{i-2}(i-2)!(n-i-m+1)!} \right)$$

Evidently we are done if the large parenthetical is equal to 1, which is equivalent to

$$\frac{2n-m-1-(m-1)}{2n-m-1} = \frac{m-1}{(2n-m-1)!} \sum_{i=2}^{n-m+1} \binom{n-1}{i-1} 2 \frac{(2i-3)!(2n-2i-m)!(n-m)!}{(i-2)!(n-i-m+1)!}$$

$$(2n-m-2)! = (m-1) \sum_{i=2}^{n-m+1} \binom{n-1}{i-1} \frac{(2i-3)!(2n-2i-m)!(n-m-1)!}{(i-2)!(n-i-m+1)!}$$

Starting the sum at $i = 1$

$$(2n-m-2)! = (m-1) \sum_{i=1}^{n-m} \binom{n-1}{i} \frac{(2i-1)!(2n-2i-m-2)!(n-m-1)!}{(i-1)!(n-i-m)!}$$

Now we break the binomial coefficient into factorials and rearrange into new binomial coefficients

$$(2n-m-2)! = (m-1) \sum_{i=1}^{n-m} \frac{(n-1)!(2i-1)!(2n-2i-m-2)!(n-m-1)!}{(n-1-i)!(i)!(i-1)!(n-i-m)!}$$

$$(2n-m-2)! = (m-1) \sum_{i=1}^{n-m} \binom{2i-1}{i} \binom{2n-2i-m-1}{n-i-1} \frac{(n-1)!(n-m-1)!}{(2n-2i-m-1)}$$

Moving $m-1$ into the sum and some factorials to the LHS

$$\binom{2n-m-2}{n-m-1} = \sum_{i=1}^{n-m} \binom{2i-1}{i} \binom{2n-2i-m-1}{n-i-1} \frac{m-1}{2n-2i-m-1}$$

$$= \sum_{i=1}^{n-m} \binom{2i-1}{i} \binom{2n-2i-m-1}{n-i-1} \left(1 - \frac{2(n-i-m)}{2n-2i-m-1}\right)$$

$$= \sum_{i=1}^{n-m} \binom{2i-1}{i} \binom{2n-2i-m-1}{n-i-1} - 2 \sum_{i=1}^{n-m-1} \binom{2i-1}{i} \binom{2n-2i-m-2}{n-i-2}$$

Note that the second sum goes to $n-m-1$ since the $i = n-m$ term gave zero. We set the sum variable to start at zero

$$= \sum_{i=0}^{n-m-1} \binom{2i+1}{i} \binom{2n-m-2-(2i+1)}{n-i} - 2 \sum_{i=0}^{n-m-2} \binom{2i+1}{i} \binom{2n-m-(2i+1)-1}{n-i-1}$$

Applying Lemma A.1 to both sums yields

$$\binom{2n-m-2}{n-m-1} = \sum_{i=0}^{n-m-1} \binom{2n-m-1}{i} - 2 \sum_{i=0}^{n-m-2} \binom{2n-m-2}{i}$$

Applying Pascal's identity to the first sum and then combining the two into a telescoping sum yields the desired result. □

We now move to the derivation of the FKT's expansion. In short, the derivation proceeds by Taylor expanding in a variable which is small for well-separated points, rearranging into a Gegenbauer expansion, and replacing the derivative term with the simpler form via the above lemma.

Theorem A.3. *If K is analytic except possibly at the origin, then for $\mathbf{r}'$,$\mathbf{r}$ within the radius of convergence,*

$$K(\|\mathbf{r}' - \mathbf{r}\|) = \sum_{k=0}^{\infty} \sum_{h \in \mathcal{H}_k} Y_k^h(\mathbf{r}) Y_k^h(\mathbf{r}')^* \mathcal{K}^{(k)}(r', r)$$

where

$$\mathcal{K}^{(k)}(r', r) \coloneqq \sum_{j=k}^{\infty} r'^j \sum_{m=1}^{j} K^{(m)}(r) r^{m-j} \mathcal{T}_{jkm}^{(\alpha)},$$

and $\mathcal{T}_{jkm}^{(\alpha)}$ are constants which depend only on the dimension and not on the kernel or data. The radius of convergence is the same as that of (6) (in the main paper).

Proof.

$$K(\|\mathbf{r}' - \mathbf{r}\|) = K(r\sqrt{1+\varepsilon})$$

Taylor expanding around $\varepsilon = 0$

$$= \sum_{n=0}^{\infty} \varepsilon^n \frac{1}{n!} \frac{\partial^n}{\partial \varepsilon^n} \left(K(r\sqrt{1+\varepsilon})\right)_{\varepsilon=0}$$

Noting $\varepsilon = \left(\frac{r'^2}{r^2} - 2\frac{r'}{r}\cos\gamma\right)$ and expanding the binomial

$$= \sum_{n=0}^{\infty} \sum_{i=0}^{n} \frac{r'^{2(n-i)}}{r^{2(n-i)}} \left(-2\frac{r'}{r}\cos\gamma\right)^i \binom{n}{i} \frac{1}{n!} \frac{\partial^n}{\partial \varepsilon^n} \left(K(r\sqrt{1+\varepsilon})\right)_{\varepsilon=0} \tag{17}$$

We will make use of

$$\cos^i\gamma = \sum_{k=0}^{i} \mathcal{A}_{ki} C_k^{\alpha}(\cos\gamma)$$

where $\alpha = d/2 - 1$, $C_k^{\alpha}(\cos\gamma)$ is the Gegenbauer polynomial, $\mathcal{A}_{ki} = 0$ when $k \neq i \mod 2$, and

$$\mathcal{A}_{ki} = \frac{i!(\alpha+k)}{2^i \frac{i-k}{2}! (\alpha)_{\frac{i+k}{2}+1}}$$

when $k = i \mod 2$. Here $(\alpha)_{\frac{i+k}{2}+1}$ denotes the rising factorial, i.e. $(\alpha)_n = (\alpha)(\alpha+1)\dots(\alpha+n-1)$. Then, substituting in for the powers of cosine in (17) yields

$$= \sum_{n=0}^{\infty}\sum_{i=0}^{n}\sum_{k=0}^{i} \mathcal{A}_{ki} C_k^{(\alpha)}(\cos\gamma) \frac{r'^{2(n-i)}}{r^{2(n-i)}} \left(-2\frac{r'}{r}\right)^i \binom{n}{i} \frac{1}{n!} \frac{\partial^n}{\partial \varepsilon^n} \left(K(r\sqrt{1+\varepsilon})\right)_{\varepsilon=0}$$

We pause to show that this triple sum is absolutely convergent. Let $\varepsilon_{\gamma=0} = \frac{r'^2}{r^2}$ be the value of ε with γ set to 0, and assume that this value is inside the radius of convergence of the above Taylor series in ε. Then

$$\sum_{n=0}^{\infty}\sum_{i=0}^{n}\sum_{k=0}^{i} \left| \mathcal{A}_{ki} C_k^{(\alpha)}(\cos\gamma) \frac{r'^{2(n-i)}}{r^{2(n-i)}} \left(-2\frac{r'}{r}\right)^i \binom{n}{i} \frac{1}{n!} \frac{\partial^n}{\partial \varepsilon^n} \left(K(r\sqrt{1+\varepsilon})\right)_{\varepsilon=0} \right|$$

$$\leq \sum_{n=0}^{\infty}\sum_{i=0}^{n} \left| \frac{r'^{2(n-i)}}{r^{2(n-i)}} \left(-2\frac{r'}{r}\right)^i \binom{n}{i} \frac{1}{n!} \frac{\partial^n}{\partial \varepsilon^n} \left(K(r\sqrt{1+\varepsilon})\right)_{\varepsilon=0} \right| \sum_{k=0}^{i} \left| \mathcal{A}_{ki} C_k^{(\alpha)}(1) \right|$$

$$= \sum_{n=0}^{\infty}\sum_{i=0}^{n} \left| \frac{r'^{2(n-i)}}{r^{2(n-i)}} \left(-2\frac{r'}{r}\right)^i \binom{n}{i} \frac{1}{n!} \frac{\partial^n}{\partial \varepsilon^n} \left(K(r\sqrt{1+\varepsilon})\right)_{\varepsilon=0} \right|$$

$$= \sum_{n=0}^{\infty} \left| \frac{\partial^n}{\partial \varepsilon^n} \left(K(r\sqrt{1+\varepsilon})\right)_{\varepsilon=0} \frac{1}{n!} \right| \sum_{i=0}^{n} \left| \frac{r'^{2(n-i)}}{r^{2(n-i)}} \left(-2\frac{r'}{r}\right)^i \binom{n}{i} \right|$$

$$= \sum_{n=0}^{\infty} \left| \frac{\partial^n}{\partial \varepsilon^n} \left(K(r\sqrt{1+\varepsilon})\right)_{\varepsilon=0} \varepsilon_{(\gamma=0)}^n \frac{1}{n!} \right|$$

Since $|\varepsilon_{\gamma=0}|$ is inside the radius of convergence of the Taylor series, then the final sum above is finite as a consequence of the Taylor series being absolutely convergent in its radius of convergence.

This absolute convergence allows us to swap the sums as we please, which we will do. First, let $j = 2n - i$ so that $i = 2n - j$, then

$$\sum_{n=0}^{\infty}\sum_{i=0}^{n}\sum_{k=0}^{i} = \sum_{n=0}^{\infty}\sum_{j=n}^{2n}\sum_{k=0}^{2n-j} = \sum_{j=0}^{\infty}\sum_{n=\lceil j/2\rceil}^{j}\sum_{k=0}^{2n-j} = \sum_{j=0}^{\infty}\sum_{k=0}^{j}\sum_{n=\frac{j+k}{2}}^{j} = \sum_{k=0}^{\infty}\sum_{j=k}^{\infty}\sum_{n=\frac{j+k}{2}}^{j}$$

So that our current form of the expansion is

$$\sum_{k=0}^{\infty}\sum_{j=k}^{\infty}\sum_{n=\frac{j+k}{2}}^{j} \mathcal{A}_{k,(2n-j)} C_k^{(\alpha)}(\cos\gamma) \frac{r'^{2(n-(2n-j))}}{r^{2(n-(2n-j))}} \left(-2\frac{r'}{r}\right)^{(2n-j)} \frac{1}{n!} \frac{\partial^n}{\partial \varepsilon^n} \left(K(r\sqrt{1+\varepsilon})\right)_{\varepsilon=0}$$

$$= \sum_{k=0}^{\infty} C_k^{(\alpha)}(\cos\gamma) \sum_{j=k}^{\infty}\sum_{n=\frac{j+k}{2}}^{j} \mathcal{A}_{k,(2n-j)} \frac{r'^{2(n-(2n-j))}}{r^{2(n-(2n-j))}} \left(-2\frac{r'}{r}\right)^{(2n-j)} \frac{1}{n!} \frac{\partial^n}{\partial \varepsilon^n} \left(K(r\sqrt{1+\varepsilon})\right)_{\varepsilon=0} \tag{18}$$

The pieces are now in place for us to arrive at the FKT's final form. Plugging (15) into (18) yields

$$\sum_{k=0}^{\infty} C_k^{(\alpha)}(\cos\gamma) \sum_{j=k}^{\infty}\sum_{n=\frac{j+k}{2}}^{j}\sum_{m=1}^{n} \mathcal{A}_{k,(2n-j)} \frac{r'^{2(n-(2n-j))}}{r^{2(n-(2n-j))}} \left(-2\frac{r'}{r}\right)^{(2n-j)} \frac{1}{n!} \mathcal{B}_{nm} K^{(m)}(r) r^m$$

$$= \sum_{k=0}^{\infty} C_k^{(\alpha)}(\cos\gamma) \sum_{j=k}^{\infty} r'^j \sum_{m=1}^{j} K^{(m)}(r) r^{m-j} \overline{\mathcal{T}}_{k,j,m}^{(\alpha)}$$

where

$$\overline{\mathcal{T}}_{k,j,m}^{(\alpha)} \coloneqq \sum_{n=\max\left(\frac{j+k}{2},m\right)}^{j} \mathcal{A}_{k,(2n-j)} (-2)^{(2n-j)} \frac{1}{n!} \mathcal{B}_{nm}$$

Finally, expanding the Gegenbauer polynomial into hyperspherical harmonics yields

$$= \sum_{k=0}^{\infty} \sum_{h \in \mathcal{H}_k} Y_k^h(\mathbf{r}) Y_k^h(\mathbf{r}')^* \sum_{j=k}^{\infty} r'^j \sum_{m=1}^{j} K^{(m)}(r) r^{m-j} \mathcal{T}_{k,j,m}^{(\alpha)}$$

□

where $\mathcal{T}_{k,j,m}^{(\alpha)} = Z_k^{(\alpha)} \overline{\mathcal{T}}_{k,j,m}^{(\alpha)}$.

A.3 NUMBER OF TERMS IN FKT

The "rank" of the low-rank expansion is given by

$$\sum_{k=0}^{P} |\mathcal{H}_k| \lfloor \frac{P-k}{2} + 1 \rfloor \tag{19}$$

where $|\mathcal{H}_k|$ is the number of linearly independent hyperspherical harmonics of order k, and $\lfloor \frac{P-k}{2} + 1 \rfloor$ is the rank of $\mathcal{K}_p^{(k)}$. The former is given by $|\mathcal{H}_k| = \binom{k+d-1}{k} - \binom{k+d-3}{k-2}$ in Wen & Avery (1985). We start by writing $\lfloor \frac{P-k}{2} + 1 \rfloor = \frac{P-k+1}{2} + \frac{1}{2}(1_{k=P \mod 2})$ and addressing the first term first.

$$\sum_{k=0}^{P} \left(\binom{k+d-1}{k} - \binom{k+d-3}{k-2} \right) \frac{P-k+1}{2}$$

$$= \sum_{k=0}^{P} \binom{k+d-1}{k} \frac{P-k+1}{2} - \sum_{k=2}^{P} \binom{k+d-3}{k-2} \frac{P-k+1}{2}$$

Further breaking apart the sum,

$$= \frac{1}{2} P \sum_{k=0}^{P} \binom{k+d-1}{k} - \frac{1}{2} \sum_{k=0}^{P} \binom{k+d-1}{k} (k-1)$$

$$- \frac{1}{2} P \sum_{k=0}^{P-2} \binom{k+d-1}{k} + \frac{1}{2} \sum_{k=0}^{P-2} \binom{k+d-1}{k} (k+1)$$

Applying the hockey stick identity yields

$$= \frac{1}{2} P \binom{d+P}{P} - \frac{1}{2} \sum_{k=0}^{P} \binom{k+d-1}{k} (k-1)$$

$$- \frac{1}{2} P \binom{d+P-2}{P-2} + \frac{1}{2} \sum_{k=0}^{P-2} \binom{k+d-1}{k} (k+1)$$

Combining the two remaining sums

$$= \frac{1}{2} P \binom{d+P}{P} - \frac{1}{2} P \binom{d+P-2}{P-2}$$

$$- \frac{1}{2}(P-2) \binom{d+P-2}{P-1} - \frac{1}{2}(P-1) \binom{d+P-1}{P} + \sum_{k=0}^{P-2} \binom{k+d-1}{k}$$

Again making use of the hockey stick identity,

$$= \frac{1}{2} P \binom{d+P}{P} - \frac{1}{2} P \binom{d+P-1}{P-1}$$

d	3	4	5	6	7	8	9
$\frac{1}{r}$	1	-	2	-	3	-	4
$\frac{1}{r^2}$	-	1	-	2	-	3	-
$\frac{1}{r^3}$	-	-	1	-	2	-	3
$\frac{1}{r}e^{-r}$	1	-	2	-	3	-	4
e^{-r}	2	-	3	-	4	-	5
re^{-r}	3	-	4	-	5	-	6
$e^{-1/r}$	4	4	4	4	4	4	4
e^{-1/r^2}	2	2	2	2	2	2	2

Table 2: For a variety of different kernels in different dimensions, the value of $\mathcal{R}_k$ achievable in (20), independent of P. Dashes indicate that $\mathcal{R}_k$ was always found to be equal to its upper bound of $\lfloor \frac{P+k-2}{2} \rfloor$. By automatically finding these shorter expressions for the radial expansions when possible, we are able to change the $\lfloor \frac{P-k+2}{2} \rfloor$ term in (19) to a constant. The 2-term radial expansion for e^{-r} is given in Table 3.

$$+\binom{d+P-2}{P-1} - \frac{1}{2}(P-1)\binom{d+P-1}{P} + \binom{d+P-2}{P-2}$$

$$= \binom{d+P-2}{P-1} + \frac{1}{2}\binom{d+P-1}{P} + \binom{d+P-2}{P-2}$$

$$= \binom{d+P-1}{P-1} + \frac{1}{2}\binom{d+P-1}{P}$$

Where we have used Pascal's identity several times. Then we address the second component,

$$\frac{1}{2} \sum_{\substack{k=0 \\ k=P \bmod 2}}^{P} \left(\binom{k+d-1}{k} - \binom{k+d-3}{k-2} \right)$$

This telescopes to

$$= \frac{1}{2}\binom{d+P-1}{P}$$

Then, summing both components up and applying Pascal's identity yields

$$\binom{d+P-1}{P-1} + \frac{1}{2}\binom{d+P-1}{P} + \frac{1}{2}\binom{d+P-1}{P} = \binom{d+P}{P}$$

A.4 COMPRESSION OF THE RADIAL EXPANSION

Here we remark on some beneficial properties of the term $\mathcal{K}^{(k)}(r', r)$ in our expansion. We define $\mathcal{R}_k$ to be the smallest number such that there exist functions $F_{k,i}, G_{k,i}$ that satisfy

$$\mathcal{K}_p^{(k)}(r', r) = \sum_{i=1}^{\mathcal{R}_k} F_{k,i}(r) G_{k,i}(r') \tag{20}$$

The motivation for focusing on this number is that it directly impacts the size $\mathcal{P}$ of our expansion, and hence the efficiency of our compression. In the case of $K(r) = 1/r$ we have $\mathcal{K}^{(k)}(r', r) = r'^k / r^{k+1}$ and so $F_{k,1}(r) = \frac{1}{r^{k+1}}, G_{k,1}(r') = r'^k$, and $\mathcal{R}_k = 1$. For general kernels, we only have $\mathcal{R}_k \leq \lfloor \frac{p-k+2}{2} \rfloor$.

However, it is possible for us to automatically detect when $F_{k,i}, G_{k,i}$ exist so that $\mathcal{R}_k$ is smaller. Consider a kernel which satisfies the differential equation $K'(r) = q(r)K(r)$, where q is a Laurent polynomial. In this case, the mth derivatives of the kernel will result in products of Laurent polynomials and the kernel itself, and hence the kernel may be pulled completely out of the double sum defining $\mathcal{K}_p^{(k)}$, yielding a binomial in r' and r

$$\mathcal{K}_p^{(k)}(r', r) = K(r) \sum_j \sum_m r'^j r^m A_{j,m},$$

$F_{k,i}(r)$		
	$i=0$	$i=1$
$k=0$	re^{-r}	$-\frac{1}{3}e^{-r}$
$k=1$	r^2e^{-r}	$e^{-r}(\frac{-1}{5}r+\frac{-1}{5})$
$k=2$	$(\frac{1}{3}r^2+\frac{1}{3}r^3)e^{-r}$	$(\frac{-1}{7}r+\frac{-1}{42}r^2+\frac{1}{42}r^3+\frac{-1}{7})e^{-r}$
$\vdots$		

$G_{k,i}(r')$		
	$i=0$	$i=1.$
$k=0$	$1+\frac{1}{6}r'^2+\frac{1}{120}r'^4+\frac{1}{5040}r'^6$	$r'^2+\frac{1}{10}r'^4+\frac{1}{280}r'^6$
$k=1$	$1+\frac{1}{10}r'^2+\frac{1}{280}r'^4+\frac{1}{15120}r'^6$	$r'^2+\frac{1}{14}r'^4+\frac{1}{504}r'^6$
$k=2$	$1+\frac{-1}{504}r'^4$	$r'^2+\frac{1}{18}r'^4$
$\vdots$		

Table 3: Note that we are using $K(r)$ as shorthand for $K(\|\mathbf{r}'-\mathbf{r}\|)$. For $K(r)=e^{-r}$ we have $\mathcal{K}^{(k)}(r,r')=F_{k,1}G_{k,1}+F_{k,2}G_{k,2}$.

where the $A_{j,m}$ coefficients are computed based on the $\mathcal{T}^{(\alpha)}_{jkm}$ terms in the FKT expansion and the coefficients of the Laurent polynomial q. The sums over j,m are finite and their range depends on the powers of the argument in the Laurent polynomial. If the $A_{j,m}$ are rational, then a concise representation of the form (20) may be found in the following way: (i) insert the coefficients $A_{j,m}$ into a matrix with rows and columns corresponding to the respective powers of r and r' in the binomial, (ii) perform a rank-revealing QR factorization (Businger & Golub, 1965; Chan, 1987) of the matrix but skip the normalization step so that all entries remain rational, and (iii) recover the functions $F_{k,i}$ from the coefficients in Q and the functions $G_{k,i}$ from the coefficients in R. Because the entries remained rational, the rank found will exactly be the sought value of $\mathcal{R}_k$.

In our implementation, we automatically perform this computation of $\mathcal{R}_k, F_{k,i}(r)$, and $G_{k,i}(r')$ when the given kernel satisfies the aforementioned differential equation (this is indicated by a user-toggled flag). Although we find $\mathcal{R}_k=\lfloor\frac{p-k+2}{2}\rfloor$ for the squared exponential, we do see significant reductions in the size of the expansion for other kernels, notably Matérn kernels. See Table 2 for some values of $\mathcal{R}_k$ for various kernels and dimensions, and Table 3 for the functions $F_{k,i}$ and $G_{k,i}$ for the exponential kernel, for which $\mathcal{R}_k=2$.

B ADDITIONAL INFORMATION FOR EXPERIMENTS

B.1 FURTHER ERROR RESULTS FOR SYNTHETIC EXPERIMENTS

We performed the accuracy measurement experiment detailed in the section of the main text concerning synthetic experiments for many kernels in many dimensions. The results are presented in Table 5. Notably the error is not significantly impacted by dimension (an observed increased accuracy with d may be due to the experimental setup exploring relatively less of the space of function arguments), and shows consistent exponential decrease with the truncation parameter p.

We also remark that oscillatory kernels are known to have higher ranks for off-diagonal blocks. In kernel-independent FMMs which use factorizations of subblocks of the matrix, the result of attempting to compress a kernel matrix whose kernel has high-frequency oscillations is that little compression is achieved, accuracy is maintained, and runtime is comparable to a dense operation. For the FKT, the result would be consistent compression and runtime, but accuracy lost (since the interactions being compressed are not low-rank, as is assumed for the method). The user may, acknowledging this behavior of the kernel matrix, increase the truncation parameter so that accuracy is maintained at the cost of runtime, but our implementation of the FKT currently has no hooks to automatically detect the need for this. A wealth of literature exists for these kernels (c.f. Cheng et al. (2006)), and it is likely that an analogous extension of the FKT to incorporate considerations present in the directional FMM would improve performance with highly oscillatory kernels.

	Dense	FKT	SKI	FITC
vs. dense		**1.0336e−4**	1.3371e−1	2.6900e−1
vs. test data	**1.1840e−1**	**1.1840e−1**	1.5727e−1	2.3858e−1

Table 4: Relative error of the GP predictive mean compared to the predictions computed without approximations (vs. dense, top row), and compared to held-out test data (vs. test data, bottom row). In order to enable comparisons to the dense case, we subsampled the oceanographic data to approximately 20 thousand points, and used an equal number of points for a held-out test set. As in the original experiments, we used a Matérn-3/2 kernel. The dense results are computed by instantiating the kernel matrix and subsequently using the method of conjugate gradients in the same way as for FKT and SKI. The number of inducing points for SKI is chosen using the `choose_grid_size` of `GPyTorch`, which creates of regular grid of n points, and for FITC is chosen to be around three thousand to achieve a comparable runtime for all accelerated methods.

Maximum Absolute Error

Kernel	$K(r) = e^{-r}$				$K(r) = \cos r/r$			
Dim.	3	6	9	12	3	6	9	12
$p = 3$	1.03e-2	1.02e-2	1.02e-2	1.02e-2	5.44e-2	3.07e-2	3.07e-2	3.06e-2
$p = 6$	7.32e-4	6.78e-4	6.52e-4	6.56e-4	7.60e-3	2.74e-3	2.01e-3	2.00e-3
$p = 9$	5.48e-5	5.47e-5	5.40e-5	5.02e-5	7.68e-4	3.65e-4	2.34e-4	1.93e-4
$p = 12$	4.62e-6	4.57e-6	4.59e-6	4.31e-6	6.03e-5	3.23e-5	3.06e-5	2.01e-5
$p = 15$	4.25e-7	4.24e-7	4.20e-7	3.98e-7	9.92e-6	3.48e-6	3.05e-6	2.59e-6
$p = 18$	4.14e-8	4.14e-8	4.04e-8	4.04e-8	1.70e-6	5.23e-7	3.12e-7	2.82e-7
Kernel	$K(r) = (1 + r^2)^{-1}$				$K(r) = e^{-r^2}$			
Dim.	3	6	9	12	3	6	9	12
$p = 3$	1.41e-2	1.41e-2	1.41e-2	1.41e-2	4.86e-2	4.27e-2	2.95e-2	2.95e-2
$p = 6$	2.17e-3	1.61e-3	1.11e-3	1.11e-3	9.42e-3	7.85e-3	4.91e-3	4.86e-3
$p = 9$	1.58e-4	1.42e-4	1.39e-4	9.51e-5	9.32e-4	5.45e-4	5.40e-4	3.87e-4
$p = 12$	1.71e-5	1.54e-5	1.19e-5	8.29e-6	4.80e-5	4.10e-5	4.10e-5	2.64e-5
$p = 15$	1.62e-6	1.27e-6	9.35e-7	9.18e-7	2.29e-6	2.29e-6	1.96e-6	1.51e-6
$p = 18$	1.39e-7	1.02e-7	7.69e-8	6.40e-8	9.88e-8	9.88e-8	6.39e-8	4.07e-8

Table 5: Experimentally observed errors which are calculated for the $p = 4$ FKT approximation by taking the maximum absolute error of the truncated expansion for 1000 randomly selected pairs of points $\mathbf{r}', \mathbf{r}$ satisfying $|\mathbf{r}'| = 1, |\mathbf{r}| = 2$.

B.2 ERROR RESULTS FOR SEA SURFACE TEMPERATURE REGRESSION EXPERIMENT

In Table B.2, we show the relative error of the GP predictive mean compared to the predictions computed without approximations (vs. dense, top row), and compared to held-out test data (vs. test data, bottom row). The FKT retains four digits of accuracy compared to the dense method, a degree of accuracy that the inducing point methods are unable to achieve. Further, the FKT attains the same relative test error as the dense method, approximately 30% lower than SKI, highlighting its performance and generality.

B.3 GAUSSIAN PROCESSES

A Gaussian Process (GP) is a distribution over functions whose finite-dimensional marginal distributions are distributed according to a multivariate normal law. That is, for any sample f of a GP, and any finite set of inputs $\mathbf{X}$, we have $f(\mathbf{X}) \sim \mathcal{N}(\boldsymbol{\mu}_\mathbf{X}, \boldsymbol{\Sigma}_\mathbf{X})$, for some mean vector $\boldsymbol{\mu}_\mathbf{X}$ and covariance matrix $\boldsymbol{\Sigma}_\mathbf{X}$. In fact, analogous to the multivariate case, a GP is completely defined by its first and second moments: a mean function $\mu(\cdot)$ and a covariance kernel $\kappa(\cdot, \cdot)$, also known as a kernel. In particular, if $f \sim \mathcal{GP}(\mu, \kappa)$ then for any finite collection of inputs $\mathbf{X}$,

$$f(\mathbf{X}) \sim \mathcal{N}(\mu(\mathbf{X}), \kappa(\mathbf{X}, \mathbf{X})), \tag{21}$$

where $\kappa(\mathbf{X}, \mathbf{X})$ is the matrix whose $(i, j)^{\text{nth}}$ entry is $\kappa(\mathbf{X}_i, \mathbf{X}_j)$. Fortunately, the posterior mean μ_p and posterior covariance κ_p of a GP conditioned on observations with normally-distributed noise have closed forms and only

require linear algebraic operations:

$$\begin{aligned}\mu_p(\mathbf{X}_*) &= \mu(\mathbf{X}_*) + \kappa(\mathbf{X}_*, \mathbf{X})\boldsymbol{\Sigma}_{\mathbf{X}}^{-1}(\mathbf{y} - \mu(\mathbf{X})),\\ \kappa_p(\mathbf{X}_*, \mathbf{X}'_*) &= \kappa(\mathbf{X}_*, \mathbf{X}'_*) - \kappa(\mathbf{X}_*, \mathbf{X})\boldsymbol{\Sigma}_{\mathbf{X}}^{-1}\kappa(\mathbf{X}, \mathbf{X}'_*),\end{aligned} \tag{22}$$

where, $\boldsymbol{\Sigma}_{\mathbf{X}} = k(\mathbf{X}, \mathbf{X}) + \sigma_y^2\mathbf{I}$ and σ_y is the standard error of the target $\mathbf{y}$. We use the first formula to calculate the predictive mean of a GP for the oceanographic data in the main text using FKT. For more background on Gaussian processes, see (Rasmussen & Williams, 2005).

C EXISTING CODES, GPU ACCELERATION, AND GP SPECIFIC IMPROVEMENTS

Deisenroth & Ng (2015) introduced the robust Bayesian Committee Machine (rBCM) which trains local GP "experts" on subsets of the data and combines their predictions. All computations of rBCM can be carried out in a distributed fashion, but no constituent model is trained on the entire data. De G. Matthews et al. (2017) introduced GPflow, a GP library based on accelerating variational inference procedures with GPUs via the TensorFlow framework. GPyTorch is also a GPU-accelerated library, but is based on PyTorch and instead of variational inference, expresses all GP inference equations via MVMs (Gardner et al., 2018b), relying on stochastic estimators to compute log-determinant and trace terms (Dong et al., 2017).