SIAM J. SCI. COMPUT.
Vol. 45, No. 5, pp. A2431–A2457

FAST EXPANSION INTO HARMONICS ON THE DISK: A STEERABLE BASIS WITH FAST RADIAL CONVOLUTIONS*

NICHOLAS F. MARSHALL†, OSCAR MICKELIN‡, AND AMIT SINGER§

Abstract. We present a fast and numerically accurate method for expanding digitized $L \times L$ images representing functions on $[-1,1]^2$ supported on the disk $\{x \in \mathbb{R}^2 : |x| < 1\}$ in the harmonics (Dirichlet Laplacian eigenfunctions) on the disk. Our method, which we refer to as the Fast Disk Harmonics Transform (FDHT), runs in $\mathcal{O}(L^2 \log L)$ operations. This basis is also known as the Fourier–Bessel basis, and it has several computational advantages: it is orthogonal, ordered by frequency, and steerable in the sense that images expanded in the basis can be rotated by applying a diagonal transform to the coefficients. Moreover, we show that convolution with radial functions can also be efficiently computed by applying a diagonal transform to the coefficients.

Key words. Laplacian eigenfunctions, steerable basis, Fourier–Bessel basis

MSC codes. 65R10, 65D18, 42-04, 33C10

DOI. 10.1137/22M1542775

1. Introduction.

1.1. Motivation.

Decomposing a function into its Fourier series can be viewed as representing a function in the eigenfunctions of the Laplacian on the torus $\mathbb{T} := [0, 2\pi]$, where 0 and 2π are identified. Indeed,

$$-\Delta e^{\imath k x} = k^2 e^{\imath k x}.$$

The eigenfunctions of the Laplacian (harmonics) on the disk $\{x \in \mathbb{R}^2 : |x| < 1\}$ that satisfy the Dirichlet boundary conditions can be written in polar coordinates $(r, \theta) \in [0, 1) \times [0, 2\pi)$ as

$$\psi_{nk}(r, \theta) = c_{nk} J_n(\lambda_{nk} r) e^{\imath n \theta}, \tag{1.1}$$

where c_{nk} is a normalization constant, J_n is the nth order Bessel function of the first kind, and λ_{nk} is the kth smallest positive root of J_n. The indices run over $(n, k) \in \mathbb{Z} \times \mathbb{Z}_{>0}$. The functions ψ_{nk} satisfy

$$-\Delta \psi_{nk} = \lambda_{nk}^2 \psi_{nk}. \tag{1.2}$$

In this paper, we present a fast and accurate transform of digitized $L \times L$ images into this eigenfunction basis often referred to as the Fourier–Bessel basis. For computational purposes, this basis is convenient for a number of reasons:

*Submitted to the journal's Methods and Algorithms for Scientific Computing section December 22, 2022; accepted for publication (in revised form) May 19, 2023; published electronically September 22, 2023.

https://doi.org/10.1137/22M1542775

Funding: The work of the first author was supported in part by NSF grant DMS-1903015. The work of the third author was supported in part by AFOSR grant FA9550-20-1-0266, the Simons Foundation Math+X Investigator Award, NSF BIGDATA Award IIS-1837992, NSF grant DMS-2009753, and NIH/NIGMS 1R01GM136780-01.

†Department of Mathematics, Oregon State University, Corvallis, OR 97330 USA (marsnich@oregonstate.edu).

‡Program in Applied and Computational Mathematics, Princeton University, Princeton, NJ 08540 USA (hm6655@princeton.edu).

§Department of Mathematics and Program in Applied and Computational Mathematics, Princeton University, Princeton, NJ 08540 USA (amits@math.princeton.edu).

(i) *Orthonormal:* These eigenfunctions are an orthonormal basis for square integrable functions on the disk.
(ii) *Ordered by frequency:* The basis functions are ordered by eigenvalues, which can be interpreted as frequencies due to the connection with the Laplacian and Fourier series described above. Low-pass filtering can be performed by retaining basis coefficients up to a given threshold.
(iii) *Steerable:* Functions expanded in the basis can be rotated by applying a diagonal transform corresponding to phase modulation of the coefficients.
(iv) *Fast radial convolutions:* We show that the convolution with radial functions can be computed by applying a diagonal transform to the coefficients.

Our Fast Disk Harmonics Transform (FDHT) method involves $\mathcal{O}(L^2 \log L)$ operations and has precise accuracy guarantees. Python code that implements our FDHT method is publicly available online.[1] To the best of our knowledge, existing methods for computing the expansion coefficients in a steerable basis [44, 45, 25, 26] either have computational complexity $\mathcal{O}(L^3)$ or suffer from low numerical precision; see section 1.4 for a more detailed discussion of past work.

Steerable bases have been utilized in numerous image-processing problems including image alignment [33], image classification [45], and image denoising [44], including applications to machine learning [5, 7, 43] and data-driven science, such as applications to cryo-electron microscopy (cryo-EM) [6, 29], and computer vision [31], among other areas.

There are many possible choices of steerable bases—for instance, Slepian functions (also known as 2D prolate spheroidal wave functions) [25, 26, 37], or Zernike polynomials which are widely used in optics [43]. The harmonics on the disk (which satisfy Dirichlet boundary conditions) [44, 45] are one natural choice due to their orthogonality, ordering by frequency, and fast radial convolution.

We illustrate the frequency ordering property of the Laplacian eigenbasis by performing a low-pass filter by projecting onto the span of eigenfunctions whose eigenvalues are below a sequence of bandlimits that decrease the number of basis functions successively by factors of four, starting from 39593 coefficients. Since the basis is orthonormal, this is equivalent to setting coefficients above the bandlimit equal to zero; see Figure 1(a)–(d). Further, we demonstrate the radial convolution property by illustrating the convolution with a point spread function, which is a function used in computational microscopy [41]; see Figure 1(e)–(f). The image used for this example is a tomographic projection of a 3D density map representing a bio-molecule (E. coli 70S ribosome) [35].

1.2. Notation. We denote the L^q-norm of a function $g : \mathbb{R}^2 \to \mathbb{C}$ and the ℓ^q-norm of a vector $v \in \mathbb{C}^d$ by $\|g\|_{L^q} := (\int_{\mathbb{R}^2} |g(x)|^q dx)^{1/q}$ and $\|v\|_{\ell^q} := (\sum_{j=1}^{d} |v_j|^q)^{1/q}$, respectively.

Let f be an $L \times L$ image whose pixel values $f_{j_1 j_2}$ are samples of a function $\tilde{f} : [-1, 1]^2 \to \mathbb{R}$ that is supported on the unit disk $\{x \in \mathbb{R}^2 : |x| < 1\}$. More precisely, we define the pixel locations by

$$x_{j_1 j_2} := (hj_1 - 1, hj_2 - 1), \quad \text{where} \quad h := 1/\lfloor (L+1)/2 \rfloor, \tag{1.3}$$

and assume the pixel values satisfy $f_{j_1 j_2} = \tilde{f}(x_{j_1 j_2})$. Let

$$x_1, \ldots, x_p \quad \text{and} \quad f_1, \ldots, f_p \tag{1.4}$$

[1] An implementation is available at https://github.com/nmarshallf/fle_2d.

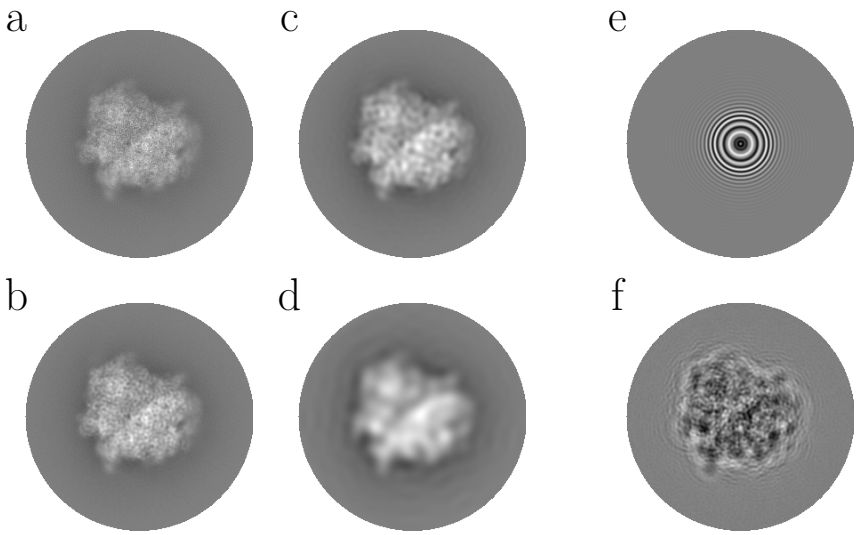

FIG. 1. *Illustration of our method for $L \times L$ images with $L = 256$. Original image* (a)*, a low-pass filter of the original image using a decreasing number of basis functions* (b)–(d)*, radial function* (e)*, convolution of the original image with radial function* (f).

denote an enumeration of the pixel locations and corresponding pixel values, respectively, where $p = L^2$ is the number of pixels in the image. For any given bandlimit $\lambda > 0$, let

$$m = \{(n,k) \in \mathbb{Z} \times \mathbb{Z}_{>0} : \lambda_{nk} \leq \lambda\} \tag{1.5}$$

denote the number of Bessel function roots (square root of eigenvalues; see (1.2)) below the bandlimit λ, and let

$$\lambda_1 \leq \cdots \leq \lambda_m \quad \text{and} \quad \psi_1, \ldots, \psi_m \tag{1.6}$$

denote an enumeration of the Bessel function roots below the bandlimit and corresponding eigenfunctions, respectively. Let

$$n_1, \ldots, n_m \quad \text{and} \quad k_1, \ldots, k_m$$

be enumerations such that $\psi_{n_j k_j} = \psi_j$. In the following, we switch between using single subscript notation (x_j, f_j, λ_j, ψ_j) and double subscript notation ($x_{j_1 j_2}$, $f_{j_1 j_2}$, λ_{nk}, ψ_{nk}) depending on which is more convenient; the choice will be clear from the context.

1.3. Main result. We consider the linear transform $B : \mathbb{C}^m \to \mathbb{C}^p$ which maps coefficients to images by

$$(B\alpha)_j = \sum_{i=1}^{m} \alpha_i \psi_i(x_j) h, \tag{1.7}$$

and its adjoint transform $B^* : \mathbb{C}^p \to \mathbb{C}^m$ which maps images to coefficients by

$$(B^* f)_i = \sum_{j=1}^{p} f_j \overline{\psi_i(x_j)} h, \tag{1.8}$$

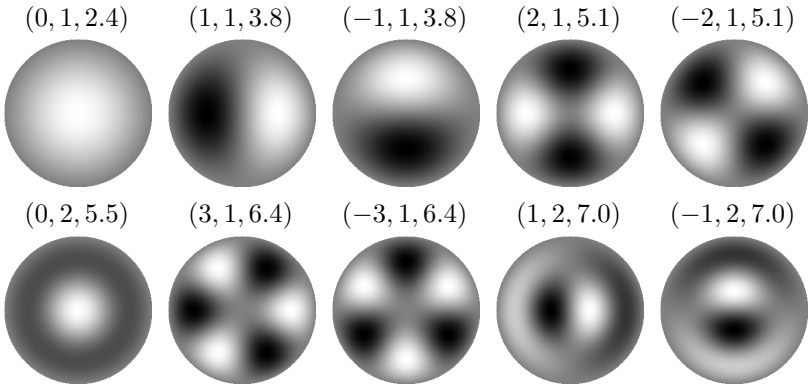

FIG. 2. *Illustration of the real version of eigenfunctions ψ_{nk} (see Remark 1.2) associated with the smallest 10 eigenvalues λ_{nk}^2. The triples above each figure show the corresponding values of (n, k, λ_{nk}), where λ_{nk} is approximated to two digits of accuracy.*

where the normalization constant h is included so that B^*B is close to the identity. Note that h^2 is the area of a pixel and thus

$$(B^*B)_{ij} = \sum_{k=1}^{p} \overline{\psi_i(x_k)}\psi_j(x_k)h^2 \tag{1.9}$$

is a Riemann sum for the integral $\int_{\mathbb{R}^2} \overline{\psi_i(x)}\psi_j(x)dx$, which equals 1 if $i = j$ and 0 otherwise. Thus, with this scaling, we expect that the entries of B^*B are close to those of the identity, at least when the number of pixels is larger than the number of basis functions.

To provide intuition about B and B^*, we visualize some of the basis functions ψ_i in Figure 2. The main result of this paper can be informally stated as follows.

THEOREM 1.1 (informal statement). *Let $\varepsilon > 0$ be any fixed accuracy, and assume $m = \mathcal{O}(p)$. Then, algorithms described in section 3.4 apply the operators $B : \mathbb{C}^m \to \mathbb{C}^p$ and $B^* : \mathbb{C}^p \to \mathbb{C}^m$ with relative error less than ε in $\mathcal{O}(p \log p)$ operations.*

Note that the statement of the theorem assumes that $m = \mathcal{O}(p)$. We make this assumption throughout the paper. Informally speaking, this means that the number of basis functions is comparable to the number of pixels in the image. The connection between the number of basis functions m, the number of pixels p, and the bandlimit parameter λ, defined in (1.5), is discussed in section 2.5. A more precise version of Theorem 1.1 is stated in section 4 (see Theorem 4.1), and supporting numerical results are reported in section 5. In particular, we show that the presented algorithm agrees with dense matrix multiplication for images with up to $p = 160^2$ pixels and accuracy parameters as small as $\varepsilon = 10^{-14}$. We report timings of images with up to $p = 512^2$ (where constructing a dense transform matrix required a prohibitive amount of memory on our testing machine). Moreover, we present a numerical example involving rotations, radial convolutions, and deconvolutions that illustrates the utility of this basis.

Remark 1.2 (real version of eigenfunctions). The complex eigenfunctions ψ_{nk} can be transformed into real eigenfunctions $\tilde{\psi}_{nk}$ via the orthogonal transformation

$$\tilde{\psi}_{0k} = \psi_{0k}, \quad \tilde{\psi}_{nk} = \frac{\psi_{nk} + (-1)^n \psi_{-nk}}{\sqrt{2}}, \quad \text{and} \quad \tilde{\psi}_{-nk} = \frac{\psi_{nk} - (-1)^n \psi_{-nk}}{i\sqrt{2}}$$

for $n \in \mathbb{Z}_{>0}$ and $k \in \mathbb{Z}_{>0}$; indeed, this follows from the definition (1.1) of ψ_{nk}, the identity $J_{-n}(r) = (-1)^n J_n(r)$, and Euler's formula $e^{ix} = \cos x + i \sin x$.

1.4. Relation to past work. In this paper, we present a fast and accurate method to apply the operators B and B^* to vectors in $\mathcal{O}(p \log p)$ operations for any fixed relative accuracy ε. We again emphasize that this is in contrast to previous results since, to the best of our knowledge, existing methods for computing the expansion coefficients in a steerable basis either require $\mathcal{O}(p^{3/2})$ operations [25, 26] or are heuristic in nature [44, 45]. Further, we mention an interesting related work [46] appearing online after our work was posted on arXiv; it follows a similar approach to this paper but considers a more general problem setting, where it achieves computational complexity $\mathcal{O}(p^{3/2})$.

The application of the operators B and B^* can be used in an iterative method to determine least-squares optimal expansion coefficients for a given image, for instance, using modified Richardson iteration or the conjugate gradient method. Alternatively, applying B^* to f can be viewed as estimating the continuous inner products that define the coefficients by using quadrature points on a grid (and potentially quadrature weights to provide an endpoint correction).

The most closely related previous approach [44, 45] expands the Fourier transform of images into the Fourier–Bessel basis by using a quadrature rule in the radial direction and an equispaced grid in the angular direction. This approach achieves complexity $\mathcal{O}(p \log p)$ but is heuristic and does not come with accuracy guarantees. Indeed, the authors of [44, 45] do not claim any accuracy guarantees, and empirically the code associated with the paper has low numerical accuracy; part of the motivation for this paper is to make a fast and accurate method that is rigorously justified and that yields a code that agrees with direct calculation to close to machine precision.

We emphasize that a key aspect of our approach is that it avoids separating variables in real space. If we were given samples of a function sampled on a polar grid instead of an image (i.e., the Cartesian grid in (1.3)), then it would be possible to separate variables and use a combination of a fast Fourier transform (FFT) and a butterfly algorithm such as [30, 39]. Our approach uses the nonuniform fast Fourier transform (NUFFT) with some classic Bessel function identities and fast polynomial interpolation to avoid numerical errors caused by performing an approximate separation in real space.

1.5. Organization. The remainder of the paper is organized as follows. In section 2, we describe the analytical apparatus underlying the method. In section 3, we describe the computational method. In section 4, we state and prove Theorem 4.1, which is a more precise version of the informal result Theorem 1.1. In section 5, we present numerical results. In section 6 we discuss the implications of the method and potential extensions.

2. Analytical apparatus.

2.1. Notation. The eigenfunctions of the Laplacian on the unit disk (that satisfy Dirichlet boundary conditions) defined in (1.1) can be extended to $\mathbb{R}^2$ as functions supported on the unit disk by

$$\psi_{nk}(r, \theta) = c_{nk} J_n(\lambda_{nk} r) e^{in\theta} \chi_{[0,1)}(r) \tag{2.1}$$

for $(n,k) \in \mathbb{Z} \times \mathbb{Z}_{>0}$, where $\chi_{[0,1)}$ denotes an indicator function for $[0,1)$. For the sake of completeness, we note that the normalization constants c_{nk} which ensure that $\|\psi_{nk}\|_{L^2} = 1$ are defined by

$$c_{nk} = \frac{1}{\pi^{1/2}|J_{n+1}(\lambda_{nk})|} \quad \text{for} \quad (n,k) \in \mathbb{Z} \times \mathbb{Z}_{>0}; \tag{2.2}$$

see [9, eq. 10.6.3, eq. 10.22.37]. We use the convention that the Fourier transform $\widehat{f} : \mathbb{R}^2 \to \mathbb{C}$ of an integrable function $f : \mathbb{R}^2 \to \mathbb{C}$ is defined by

$$\widehat{f}(\xi) = \frac{1}{2\pi} \int_{\mathbb{R}^2} f(x) e^{-\imath x \cdot \xi} dx, \tag{2.3}$$

where $x \cdot \xi$ denotes the Euclidean inner product. We define the convolution of two functions $f, g : \mathbb{R}^2 \to \mathbb{C}$ by

$$(f * g)(x) = \int_{\mathbb{R}^2} f(x-y) g(y) dy.$$

Furthermore, we will make use of the identity

$$J_n(r) = \frac{1}{2\pi} \int_0^{2\pi} e^{\imath r \sin\theta} e^{-\imath n\theta} d\theta; \tag{2.4}$$

see, for example, [38, eq. 9.19]. We note that the identities derived in the subsequent sections are similar to those derived in [17, 16, 15, 18].

2.2. Fourier transform of eigenfunctions. The analytic foundation for the presented fast method is the following expression for the Fourier transform of the functions ψ_{nk} defined in (2.1), which we prove for completeness.

LEMMA 2.1. *The Fourier transform $\widehat{\psi}_{nk}$ can be expressed by*

$$\widehat{\psi}_{nk}(\xi) = (-\imath)^n e^{\imath n\phi} \int_0^1 c_{nk} J_n(\lambda_{nk} r) J_n(\rho r) r dr, \tag{2.5}$$

where (ρ, ϕ) are polar coordinates for $\xi = (\rho\cos\phi, \rho\sin\phi)$.

Proof of Lemma 2.1. By the definition of the Fourier transform (2.3) we have

$$\widehat{\psi}_{nk}(\xi) = \frac{1}{2\pi} \int_{\mathbb{R}^2} \psi_{nk}(x) e^{-\imath x \cdot \xi} dx.$$

Changing to polar coordinates $\xi = (\rho\cos\phi, \rho\sin\phi)$ and $x = (r\cos\theta, r\sin\theta)$ gives

$$\widehat{\psi}_{nk}(\xi) = \frac{1}{2\pi} \int_0^{2\pi} \int_0^1 c_{nk} J_n(\lambda_{nk} r) e^{\imath n\theta} e^{-\imath r\rho\cos(\theta-\phi)} r dr d\theta,$$

where we used the fact that $x \cdot \xi = r\rho\cos(\theta - \phi)$. Changing variables $\theta \mapsto -\theta + \phi - \pi/2$ and taking the integral over θ gives

$$\widehat{\psi}_{nk}(\xi) = (-\imath)^n e^{\imath n\phi} \int_0^1 c_{nk} J_n(\lambda_{nk} r) J_n(\rho r) r dr,$$

as desired. ∎

2.3. Coefficients from eigenfunction Fourier transform. Next, we observe how the coefficients of a function in the eigenfunction basis can be computed by an application of Lemma 2.1. In the following, we will write the arguments of Fourier transforms of functions in polar coordinates (ρ, ϕ). We have the following result.

LEMMA 2.2. *Suppose that $\mathcal{I} \subset \mathbb{Z} \times \mathbb{Z}_{>0}$ is a finite index set, and set*

$$f = \sum_{(n,k)\in\mathcal{I}} \alpha_{nk}\psi_{nk}, \tag{2.6}$$

where $\alpha_{nk} \in \mathbb{C}$ are coefficients. Define $\beta_n : [0,\infty) \to \mathbb{C}$ by

$$\beta_n(\rho) := \imath^n \int_0^{2\pi} \widehat{f}(\rho,\phi) e^{-\imath n\phi} d\phi. \tag{2.7}$$

It then holds that

$$\alpha_{nk} = c_{nk}\beta_n(\lambda_{nk}). \tag{2.8}$$

The proof is a direct consequence of Lemma 2.1.

Proof of Lemma 2.2. Observe that (2.5) implies

$$\imath^n \int_0^{2\pi} \widehat{\psi}_{n'k'}(\rho,\phi) e^{-\imath n\phi} d\phi = 2\pi\delta_{n,n'} \int_0^1 c_{n'k'} J_{n'}(\lambda_{n'k'} r) J_{n'}(\rho r) r dr, \tag{2.9}$$

where $\delta_{n,n'} = 1$ if $n = n'$ and $\delta_{n,n'} = 0$ otherwise. Evaluating (2.9) at radius $\rho = \lambda_{nk}$ gives

$$\begin{aligned}\imath^n \int_0^{2\pi} \widehat{\psi}_{n'k'}(\lambda_{nk},\phi) e^{-\imath n\phi} d\phi &= 2\pi\delta_{n,n'} \int_0^1 c_{n'k'} J_{n'}(\lambda_{n'k'} r) J_{n'}(\lambda_{nk} r) r dr \\ &= 2\pi\delta_{n,n'} \int_0^1 c_{nk'} J_n(\lambda_{nk'} r) J_n(\lambda_{nk} r) r dr = \frac{1}{c_{nk}} \delta_{n,n'}\delta_{k,k'},\end{aligned}$$

where the final equality follows from the orthogonality of the eigenfunctions $\psi_{nk'}$ (which is a consequence of the fact that the Laplacian is self-adjoint). By the definition of β_n in (2.7), this implies that

$$\beta_n(\lambda_{nk}) = \sum_{(n',k')\in\mathcal{I}} \alpha_{n'k'} \imath^n \int_0^{2\pi} \widehat{\psi}_{n'k'}(\lambda_{nk},\phi) e^{-\imath n\phi} d\phi = \sum_{(n',k')\in\mathcal{I}} \frac{\alpha_{n'k'}}{c_{n'k'}} \delta_{n,n'}\delta_{k,k'} = \frac{\alpha_{nk}}{c_{nk}},$$

which concludes the proof. ∎

Remark 2.3 (special property of Bessel functions). We emphasize that the integral expression (2.4) of the Bessel function is crucial for the fast method of this paper. The possibility of extending the approach to create other fast transforms defined on domains in $\mathbb{R}^2$ therefore hinges on identifying equally useful integral expressions for the corresponding transforms.

2.4. Convolution with radial functions. Let $g(x) = g(|x|)$ be a radial function. In this section, we observe how the convolution with g can be computed via a diagonal transform of the coefficients. More precisely, we compute the projection of the convolution with g onto the span of any finite basis of the eigenfunctions ψ_{nk}.

LEMMA 2.4. *Let f be a function with coefficients α_{nk} as in* (2.6)*, and let $g(x) = g(|x|)$ be a radial function. We have*

$$P_{\mathcal{I}}(f * g) = \sum_{(n,k)\in\mathcal{I}} \alpha_{nk}\widehat{g}(\lambda_{nk})\psi_{nk},$$

where $P_{\mathcal{I}}$ denotes the orthogonal projection onto the span of $\{\psi_{nk}\}_{(n,k)\in\mathcal{I}}$.

The proof is a direct application of Lemma 2.2.

Proof of Lemma 2.4. We use the notation $g(x) = g(|x|)$ and $\widehat{g}(\xi) = \widehat{g}(|\xi|)$. Since the functions ψ_{nk} are an orthonormal basis, in order to compute the orthogonal projection $P_{\mathcal{I}}$, it suffices to determine the coefficients of $f * g$ with respect to ψ_{nk} for $(n,k) \in \mathcal{I}$. Since $\widehat{(f * g)}(\rho,\phi) = \widehat{f}(\rho,\phi)\widehat{g}(\rho)$, and $\widehat{g}$ is radial, we have

$$\imath^n \int_0^{2\pi} \widehat{(f * g)}(\lambda_{nk},\phi)e^{-\imath n\phi}d\phi = \imath^n \int_0^{2\pi} \widehat{f}(\lambda_{nk},\phi)\widehat{g}(\lambda_{nk})e^{-\imath n\phi}d\phi = \frac{\alpha_{nk}}{c_{nk}}\widehat{g}(\lambda_{nk}),$$

where the final equality follows from (2.8). An application of Lemma 2.2 then completes the proof. ∎

2.5. Maximum bandlimit. In this section, we use Weyl's law and lattice point counting estimates to derive a bound on the bandlimit parameter λ in terms of the number of pixels p under the assumption that the number of basis functions should not exceed the number of pixels corresponding to points in the unit disk.

Recall from (1.5) that the number of basis functions m is determined from λ by

$$m = \#\{(n,k) \in \mathbb{Z} \times \mathbb{Z}_{>0} : \lambda_{nk} \le \lambda\},$$

where λ_{nk} is the kth smallest positive root of J_n. Further, recall that λ_{nk}^2 are the eigenvalues of the Dirichlet Laplacian on the unit disk; see (1.2). Thus, it follows from Weyl's law that

$$\#\{(n,k) \in \mathbb{Z} \times \mathbb{Z}_{>0} : \lambda_{nk} \le \lambda\} = \frac{\lambda^2}{4} - \frac{\lambda}{2} + \mathcal{O}(\lambda^{2/3}); \tag{2.10}$$

see [8]. On the other hand, the number of pixels representing points in the unit disk is equal to the number of integer lattice points from $\mathbb{Z}^2$ inside a disk of radius $\lfloor(\sqrt{p}+1)/2\rfloor$; see (1.3). Classic lattice point counting results give

$$\#\left\{(j_1,j_2) \in \mathbb{Z} \times \mathbb{Z} : j_1^2 + j_2^2 \le \left\lfloor \frac{\sqrt{p}+1}{2} \right\rfloor^2 \right\} = \pi \left\lfloor \frac{\sqrt{p}+1}{2} \right\rfloor^2 + \mathcal{O}(p^{1/3}); \tag{2.11}$$

see, for example, [14, 23, 40]. Equating (2.10) with (2.11) results in

$$\lambda = 2\sqrt{\pi}\left\lfloor \frac{\sqrt{p}+1}{2} \right\rfloor + 1 + \mathcal{O}\left(p^{-1/6}\right). \tag{2.12}$$

For simplicity, motivated by (2.12), we assume

$$\lambda \le \sqrt{\pi p}. \tag{2.13}$$

Practically speaking, it can be advantageous to expand the image using fewer basis functions than described by (2.12). See, for example, the heuristic described by Remark 5.1, or see [44].

3. Computational method. In this section, we describe how to apply the operators B and B^* defined above in section 1.3 in $\mathcal{O}(p \log p)$ operations. For the purpose of exposition, we start by describing a simplified method before presenting the full method. The section is organized as follows:

- In section 3.1, we introduce notation for the algorithm description.
- In section 3.2, we give an informal description of a simplified method to apply B and B^* in $\mathcal{O}(p^{3/2} \log p)$ operations. The simplified method is a direct application of the lemmas from the previous section.
- In section 3.3, we provide an informal description of how to modify the simplified method to create a fast method to apply B and B^* in $\mathcal{O}(p \log p)$ operations. The main additional ingredient is a fast method of interpolation from Chebyshev nodes.
- In section 3.4, we give a detailed description of the fast method to apply B and B^* in $\mathcal{O}(p \log p)$ operations.

3.1. Notation. Recall that $x_1, \ldots, x_p$ and $f_1, \ldots, f_p$ are an enumeration of the pixel locations and corresponding pixel values, and $\lambda_1, \ldots, \lambda_m$ and $\psi_1, \ldots, \psi_m$ are an enumeration of the Bessel function roots and corresponding eigenfunctions; see section 1.2. Let $c_1, \ldots, c_m$ be an enumeration of the normalization constants defined in (2.2) such that c_j is the normalization constant associated with ψ_j, and let $n_1, \ldots, n_m$ and $k_1, \ldots, k_m$, be an enumeration of the Bessel function orders and root numbers such that $\psi_{n_j k_j} = \psi_j$. Further, we define $N_m = \max\{n_j \in \mathbb{Z} : j \in \{1, \ldots, p\}\}$ to be the maximum order of the Bessel functions, and $K_n := \max\{k \in \mathbb{Z}_{>0} : \lambda_{nk} \leq \lambda\}$ for $n \in \{-N_m, \ldots, N_m\}$.

A key ingredient in the simplified and fast methods is the nonuniform fast Fourier transform (NUFFT) [11, 21, 28], which is now a standard tool in computational mathematics. Given n source points and m target points in $\mathbb{R}^d$, and $1 \geq \varepsilon > 0$, the NUFFT involves

$$\mathcal{O}\left(n \log n + m \left(\log \frac{1}{\varepsilon}\right)^d\right) \tag{3.1}$$

operations to achieve ℓ^1-ℓ^∞ relative error ε; see [1, eq. (9)] and [2, sec. 1.1]. Throughout the paper (except for Theorem 4.1 and its proof), we treat ε as a fixed constant, say, $\varepsilon = 10^{-7}$, and do not include it in computational complexity statements.

3.2. Informal description of simplified method. In this section, we present a simplified method that applies B and B^* in $\mathcal{O}(p^{3/2} \log p)$ operations. We first describe how to apply B^*. The basic idea is to apply Lemma 2.2 to the function

$$f(x) = \sum_{j=1}^{p} f_j \delta(x - x_j),$$

where δ is a Dirac delta distribution. Observe that, by our convention (2.3), the Fourier transform of f is

$$\widehat{f}(\xi) = \frac{1}{2\pi} \sum_{j=1}^{p} f_j e^{-\imath x_j \cdot \xi}.$$

In polar coordinates $x_j = (r_j \cos\theta_j, r_j \sin\theta_j)$ and $\xi = (\rho \cos\phi, \rho \sin\phi)$

$$\widehat{f}(\rho, \phi) = \frac{1}{2\pi} \sum_{j=1}^{p} f_j e^{-\imath r_j \rho \cos(\theta_j - \phi)},$$

and by the definition (2.7) of β_n we have

$$\beta_n(\rho) = \sum_{j=1}^{p} f_j \frac{\imath^n}{2\pi} \int_0^{2\pi} e^{-\imath r_j \rho \cos(\theta_j - \phi)} e^{-\imath n \phi} d\phi. \tag{3.2}$$

Changing variables $\phi \mapsto \phi + \theta_j + \pi/2$ and using the identity (2.4) gives

$$\beta_n(\rho) = \sum_{j=1}^{p} f_j J_n(r_j \rho) e^{-\imath n \theta_j}.$$

By the definition (1.8) of B^* it follows that

$$(B^* f)_i = c_i \beta_{n_i}(\lambda_i) h.$$

In order to implement the above calculations numerically, we need to discretize the integral in (3.2). In Lemma 4.2, we prove that discretizing ϕ using $s = \mathcal{O}(\sqrt{p})$ equispaced angles guarantees that sums over the equispaced angles approximate integrals over ϕ to sufficient accuracy. In more detail, the simplified method for applying B^* can be described as follows.

Step 1. Using the type-2 2D NUFFT compute

$$a_{i\ell} := \sum_{j=1}^{p} f_j e^{-\imath x_j \cdot \xi_{i\ell}}, \quad \text{where} \quad \xi_{i\ell} := \lambda_i(\cos\phi_\ell, \sin\phi_\ell),$$

for $(i, \ell) \in \{1, \ldots, m\} \times \{0, \ldots, s-1\}$, where $\phi_\ell = 2\pi\ell/s$. The computational complexity of this step is $\mathcal{O}(p^{3/2})$ using the NUFFT since there are $\mathcal{O}(p)$ source nodes and $\mathcal{O}(p^{3/2})$ target nodes; see (3.1).

Step 2. Using the FFT compute

$$\beta_n(\lambda_i) \approx \frac{\imath^n}{s} \sum_{\ell=0}^{s-1} a_{i\ell} e^{-\imath n \phi_\ell}$$

for $(i, n) \in \{1, \ldots, m\} \times \{0, \ldots, s-1\}$. Since this step involves $m = \mathcal{O}(p)$ FFTs of size $s = \mathcal{O}(\sqrt{p})$, the computational complexity of this step is $\mathcal{O}(p^{3/2} \log p)$.

Step 3. By Lemma 2.2, it follows that

$$(B^* f)_i = \beta_{n_i}(\lambda_i) c_i h$$

for $i \in \{1, \ldots, m\}$. The computational complexity of this step is $\mathcal{O}(p)$ since it only involves selecting and scaling $\beta_{n_i}(\lambda_i)$ and since $m = \mathcal{O}(p)$.

3.3. Sketch of fast method. In this section, we describe how the computational complexity of the simplified method of the previous section for applying B^* can be improved from $\mathcal{O}(p^{3/2} \log p)$ to $\mathcal{O}(p \log p)$ by using fast interpolation from Chebyshev nodes.

The problem with the simplified method is the first step: it involves computing $\widehat{f}(\rho, \phi)$ for $\mathcal{O}(p)$ values of ρ and $\mathcal{O}(\sqrt{p})$ values of ϕ for a total of $\mathcal{O}(p^{3/2})$ points, which is already prohibitively expensive. Fortunately, there is a simple potential solution to this problem: since the functions $\beta_n(\rho)$ are analytic functions of ρ, it is possible to tabulate them at appropriate points and then use polynomial interpolation to compute the coefficients. We take this approach to design a fast method. Crucially, we prove

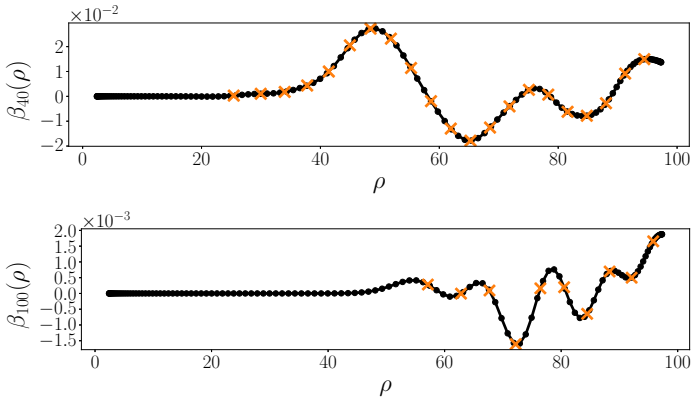

FIG. 3. *We visualize the interpolation step for a* 64×64 *input image. For* $n = 40$ *and* 100 *we plot* $\beta_n(\rho)$ *(black line), interpolation source nodes (black dots), and target points (orange crosses). (Color available online.)*

that tabulating each β_n at $\mathcal{O}(\sqrt{p})$ Chebyshev nodes is sufficient to achieve the desired accuracy (see Lemma 4.3). This reduces the number of target points in the NUFFT in the first step of the algorithm from $\mathcal{O}(p^{3/2})$ to $\mathcal{O}(p)$. Note that one should not expect to be able to use $o(p)$ points in total, since the images have p pixels.

In more detail, here is an informal summary of the fast method:

- Compute the Fourier transform of f at
$$\xi_{k\ell} := t_k(\cos\phi_\ell, \sin\phi_\ell)$$
for $\mathcal{O}(\sqrt{p})$ Chebyshev nodes t_k and $\mathcal{O}(\sqrt{p})$ angles ϕ_ℓ.
- Approximate $\beta_n(t_k)$ for the $\mathcal{O}(\sqrt{p})$ Chebyshev nodes t_k in the interval $[\lambda_1, \lambda_m]$ and $\mathcal{O}(\sqrt{p})$ frequencies n.
- For each of the $\mathcal{O}(\sqrt{p})$ frequencies, use fast interpolation from the $\mathcal{O}(\sqrt{p})$ Chebyshev nodes t_k to the $\mathcal{O}(\sqrt{p})$ Bessel function roots associated with each frequency n. We illustrate the interpolation step in Figure 3.

3.4. Detailed description of fast method. In addition to the notation of section 3.1, let

$$t_k := \frac{\lambda_m - \lambda_1}{2} \cos\left(\frac{2k+1}{q} \cdot \frac{\pi}{2}\right) + \frac{\lambda_1 + \lambda_m}{2}, \quad k = 0, \ldots, q-1, \tag{3.3}$$

be Chebyshev nodes of the first kind in the interval $[\lambda_1, \lambda_m]$ for fixed integer q. We present a detailed description of the fast method for applying B^* in Algorithm 3.1.

Remark 3.1 (methods for fast interpolation from Chebyshev nodes). Given values $v_0, \ldots, v_{q-1}$, we denote by P the $q-1$ degree polynomial such that $P(t_k) = v_k$ for $k \in \{0, \ldots, q-1\}$, where t_k are the Chebyshev nodes defined in (3.3). We can explicitly write P as

$$P(t) = \sum_{k=0}^{q-1} v_k u_k(t), \quad \text{where} \quad u_k(t) = \frac{\prod_{\ell \neq k}(t - t_\ell)}{\prod_{\ell \neq k}(t_k - t_\ell)}, \text{ for } k \in \{0, \ldots, q-1\}. \tag{3.4}$$

Given r target points $w_0, \ldots, w_{r-1}$, the map $(v_0, \ldots, v_{q-1}) \mapsto (P(w_0), \ldots, P(w_{r-1}))$ is a linear mapping $\mathbb{C}^q \to \mathbb{C}^r$. This linear operator (and its adjoint) can be applied

Algorithm 3.1. Fast method for applying B^*.

Input: Image f, bandlimit λ, and accuracy parameter ε.

Constants: # of pixels p, # of basis functions m, ε^{dis} defined by (A.9), ε^{nuf} and ε^{fst} defined by (A.6), $s = \lceil \max\{7.09\sqrt{p}, |\log_2 \varepsilon^{\text{dis}}|\}\rceil$, and $q = \lceil \max\{2.4\sqrt{p}, |\log_2 \varepsilon^{\text{dis}}|\}\rceil$.

Output: α approximating $B^* f$ to relative error ε (see Theorem 4.1).

1 Using the type-2 NUFFT, compute

$$a_{k\ell} := \sum_{j=1}^{p} f_j e^{-\imath x_j \cdot \xi_{k\ell}},$$

with relative error ε^{nuf}, where $\xi_{k\ell} := t_k(\cos\phi_\ell, \sin\phi_\ell)$ and $\phi_\ell = 2\pi\ell/s$, for $k \in \{0, \ldots, q-1\}$ and $\ell \in \{0, \ldots, s-1\}$.

2 Using FFT, compute

$$\beta_{nk} := \frac{\imath^n}{s} \sum_{\ell=0}^{s-1} a_{k\ell} e^{-\imath n \phi_\ell},$$

for $k \in \{0, \ldots, q-1\}$ and $n \in \{-N_m, \ldots, N_m\}$.

3 Using fast Chebyshev interpolation, compute

$$\alpha_i := \sum_{k=0}^{q-1} c_i \beta_{n_i k} u_k(\lambda_i) h,$$

for $i \in \{1, \ldots, m\}$ with relative error ε^{fst} for $k \in \{0, \ldots, q-1\}$ and $n \in \{-N_m, \ldots, N_m\}$, where $u_k(t)$ is defined in (3.4).

quickly by a variety of methods: for example, the interpolation could be performed by using the NUFFT [11, 21, 28] in $\mathcal{O}(p \log p)$ operations (which is often called spectral interpolation), the Fast Multipole Method (FMM) [10] in $\mathcal{O}(p)$ operations, or generalized Gaussian quadrature [19] in $\mathcal{O}(p \log p)$ operations. See also [32, Chapter 7.4] for an approach using the nonuniform fast discrete cosine transform. Although the FMM has the lowest computational complexity, it is known to have a large run-time constant, so using other methods may be faster in applications. Practically speaking, choosing a fixed number of source points centered around each target point (say 20 source points) and then applying a precomputed sparse (barycentric interpolation [3]) matrix may be more practical than any of these methods; sparse interpolation can be used in combination with spectral interpolation (by discrete cosine transform) to first increase the number of Chebyshev nodes.

Algorithm 3.2 details the fast method for applying B, which consists of applying the adjoint of the operator applied in each step of Algorithm 3.1 in reverse order, with slightly different accuracy parameters. Indeed, each step of Algorithm 3.1 consists of applying a linear transform whose adjoint can be applied in a similar number of operations: the adjoint of the first step (which uses type-2 NUFFT) is a type-1 2D NUFFT [1], the adjoint of the second step (which uses a standard FFT) is an inverse FFT, and the adjoint of the third step (fast interpolation, see Remark 3.1) can be computed by a variety of methods (including NUFFT).

Algorithm 3.2. Fast method for applying B.

Input: Coefficients α, bandlimit λ, and accuracy parameter ε.

Constants: # of pixels p, # of basis functions m, ε^{dis} defined by (A.15), ε^{fst} and ε^{nuf} defined by (A.13), $s = \lceil \max\{7.09\sqrt{p}, |\log_2 \varepsilon^{\text{dis}}|\} \rceil$, and $q = \lceil \max\{2.4\sqrt{p}, |\log_2 \varepsilon^{\text{dis}}|\} \rceil$.

Output: f approximating $B\alpha$ to relative error ε (see Theorem 4.1)

1 Using a fast Chebyshev interpolation method, compute

$$\beta^*_{nk} = h \sum_{i:n_i=n} u_k(\lambda_i) c_i \alpha_i,$$

with relative error less than ε^{fst} for $n \in \{-N_m, \dots, N_m\}$ and $k \in \{0, \dots, q-1\}$, where $u_k(t)$ is defined in (3.4).

2 Using FFT compute

$$a^*_{k\ell} := \sum_{n=-N_m}^{N_m} \frac{(-\imath)^n}{s} e^{\imath n \phi_\ell} \beta^*_{nk},$$

for all $k \in \{0, \dots, q-1\}$ and $\ell \in \{0, \dots, s-1\}$, where $\phi_\ell = 2\pi\ell/s$.

3 Using the type-1 NUFFT compute

$$f_j = \sum_{k=0}^{q-1} \sum_{\ell=0}^{s-1} e^{\imath x_j \cdot \xi_{k\ell}} a^*_{i\ell},$$

with relative error less than ε^{nuf}, for $j \in \{1, \dots, p\}$, where $\xi_{k\ell} := t_k(\cos\phi_\ell, \sin\phi_\ell)$.

4. Accuracy guarantees for the fast methods. We state and prove a precise version of the informal result in Theorem 1.1.

THEOREM 4.1. *Let $1 \geq \varepsilon > 0$ be given, and assume $\lambda \leq \sqrt{\pi p}$ and $|\log\varepsilon| \leq \sqrt{p}$. Let $\tilde{B}^*$ and $\tilde{B}$ be operators whose actions consist of applying Algorithm* 3.1 *and Algorithm* 3.2*, respectively. We have*

$$\|\tilde{B}^* f - B^* f\|_{\ell^\infty} \leq \varepsilon \|f\|_{\ell^1} \quad \text{and} \quad \|\tilde{B}\alpha - B\alpha\|_{\ell^\infty} \leq \varepsilon \|\alpha\|_{\ell^1}.$$

Moreover, both algorithms involve $\mathcal{O}(p \log p + p|\log\varepsilon|^2)$ operations.

The proof of Theorem 4.1 is given in Appendix A. We note that the theorem quantifies the computational accuracy in terms of ℓ^1-ℓ^∞ relative error, which is standard for algorithms involving the NUFFT [1, 2]. The assumption $|\log\varepsilon| \leq \sqrt{p}$ is not restrictive since if $|\log\varepsilon| \geq \sqrt{p}$, then we could directly evaluate $B^* f$ in the same asymptotic complexity $\mathcal{O}(p^2)$. The proof of Theorem 4.1 relies on the following two key lemmas that estimate a sufficient number of angular nodes and radial nodes in sections 4.1 and 4.2, respectively.

4.1. Number of angular nodes. Informally speaking, the following lemma shows that $s = \mathcal{O}(\sqrt{p})$ angular nodes are sufficient to achieve error γ in the discretization of the integral over ϕ; see (3.2).

LEMMA 4.2. *Let the number of equispaced angular nodes s satisfy*

$$s = \lceil \max\{7.09\sqrt{p}, \log_2 \gamma^{-1}\} \rceil. \tag{4.1}$$

Let $x_j = (r_j \cos\theta_j, r_j \sin\theta_j)$. *If* $\rho \in [\lambda_1, \lambda_m]$, *then*

$$\left| \frac{i^n}{s} \sum_{\ell=0}^{s-1} e^{\imath r_j \rho \cos(\theta_j - \phi_\ell)} e^{-\imath n \phi_\ell} - J_n(r_j \rho) e^{-\imath n \theta_j} \right| \leq \gamma$$

for $n \in \{-N_m, \ldots, N_m\}$ *and* $j \in \{1, \ldots, p\}$, *where* $\phi_\ell = 2\pi\ell/s$.

It will be clear from the proof that the constant 7.09 in the statement of the lemma is an overestimate; see Remark 4.4 for a discussion of how this constant can be improved.

Proof of Lemma 4.2. Let

$$g_{nj}(\rho, \phi) = \frac{i^n}{2\pi} e^{\imath r_j \rho \cos(\theta_j - \phi)} e^{-\imath n \phi} \tag{4.2}$$

for $n \in \{0, \ldots, s-1\}$ and $j \in \{1, \ldots, p\}$. We want to show that

$$\left| \frac{2\pi}{s} \sum_{\ell=0}^{s-1} g_{nj}(\rho, \phi_\ell) - J_n(r_j \rho) e^{-\imath n \theta_j} \right| < \gamma. \tag{4.3}$$

Notice that the sum in (4.3) is a discretization of the integral

$$\int_0^{2\pi} g_{nj}(t_k, \phi) d\phi = J_n(r_j t_k) e^{-\imath n \theta_j}, \tag{4.4}$$

where the exact expression for the integral results from (2.4) and a change of variables from $\phi \mapsto \phi + \theta_j + \pi/2$ in the integral. It follows from Lemma A.3 that

$$\begin{aligned} \left| \frac{2\pi}{s} \sum_{\ell=0}^{s-1} g_{nj}(\rho, \phi_\ell) - J_n(r_j \rho) e^{-\imath n \theta_j} \right| &= \left| \frac{2\pi}{s} \sum_{\ell=0}^{s-1} g_{nj}(\rho, \phi_\ell) - \int_0^{2\pi} g_{nj}(\rho, \phi) d\phi \right| \\ &\leq \frac{4 \| g_{nj}^{(s)}(\rho, \cdot) \|_{L^1}}{s^s}, \end{aligned} \tag{4.5}$$

where $g_{nj}^{(s)}(\rho, \phi)$ denotes the sth derivative of $g_{nj}(\rho, \phi)$ with respect to ϕ. From definition (4.2) of $g_{nj}(\rho, \phi)$, we have the estimate

$$\left| g_{nj}^{(s)}(\rho, \phi) \right| \leq \frac{(\lambda_m + N_m)^s}{2\pi}$$

for all $\rho \in [\lambda_1, \lambda_m]$, $n \in \{-N_m, \ldots, N_m\}$, $j = 1, \ldots, p$, and $\phi \in [0, 2\pi]$. Therefore, since $4/(2\pi) \leq 1$, it suffices to choose s such that

$$\left(\frac{\lambda_m + N_m}{s} \right)^s \leq \gamma. \tag{4.6}$$

It follows that choosing $s = \max\{2(\lambda_m + N_m), \log_2 \gamma^{-1}\}$ achieves error at most γ. To complete the proof, we note that $\lambda_m \leq \lambda$, where λ is the maximum bandlimit from section 2.5. Also by [9, 10.21.40] we have

$$\lambda_{n1} = n + 1.8575 n^{1/3} + \mathcal{O}(n^{-1/3}),$$

which implies that the maximum angular frequency

$$N_m \leq \lambda. \tag{4.7}$$

We conclude that $s = \max\{4\lambda, \log_2 \gamma^{-1}\}$ is sufficient to achieve error γ. Since we assume $\lambda \leq \sqrt{\pi p}$ and $4\sqrt{\pi} \leq 7.09$, the proof is complete. ∎

4.2. Number of radial nodes. The following lemma shows that $\mathcal{O}(\sqrt{p})$ Chebyshev nodes are sufficient for accurate interpolation in Step 3 of Algorithm 3.1.

LEMMA 4.3. *Let the number of radial nodes*

$$q = \lceil \max\left\{2.4\sqrt{p}, \log_2 \gamma^{-1}\right\} \rceil. \tag{4.8}$$

Let P_n be the degree $q-1$ polynomial such that

$$P_n(t_k) = J_n(r_j t_k) e^{-\imath n \theta_j}$$

for $k \in \{0, \ldots, q-1\}$, where t_k are Chebyshev nodes for $[\lambda_1, \lambda_m]$; see (3.3)*. Then,*

$$|P_n(\rho) - J_n(r_j \rho) e^{-\imath n \theta_j}| \leq \gamma$$

for $\rho \in [\lambda_1, \lambda_m]$, $n \in \{-N_m, \ldots, N_m\}$, and $j \in \{1, \ldots, p\}$.

As above, we emphasize that the constant 2.4 in the statement of this result is an overestimate. See Remark 4.4 for a discussion about how this constant can be improved.

Proof of Lemma 4.3. When interpolating a smooth differentiable function h defined on the interval $[a, b]$ using an interpolating polynomial P at q Chebyshev nodes, the residual term $R(\rho) = h(\rho) - P(\rho)$ can be written as

$$|R(\rho)| \leq \frac{C_q}{q!} \left(\frac{b-a}{4}\right)^q,$$

where $C_q := \max_{\rho \in [a,b]} |h^{(q)}(\rho)|$; see [34, Lemma 2.1]. If we apply this result with $[a, b] = [\lambda_1, \lambda_m]$, the residual satisfies

$$|R(\rho)| \leq \frac{C_q}{q!} \left(\frac{\lambda_m - \lambda_1}{4}\right)^q \leq \frac{C_q}{q!} \left(\frac{\sqrt{\pi p}}{4}\right)^q,$$

where the final inequality follows from the bound $\lambda_m \leq \sqrt{\pi p}$; see section 2.5. In order to apply this bound to $J_n(r_j \rho) e^{-\imath n \theta_j}$, we estimate

$$C_q := \max_{\rho \in [\lambda_1, \lambda_m]} \left| \frac{\partial^q}{\partial \rho^q} \left(J_n(r_j \rho)\right) e^{-\imath n \theta_j} \right|.$$

We expand the function $J_n(r_j \rho)$ using the integral identity in (2.4) and obtain

$$\begin{aligned} \left| \frac{\partial^q}{\partial \rho^q} \left(J_n(r_j \rho)\right) \right| &= \left| \frac{1}{2\pi} \int_0^{2\pi} \frac{\partial^q}{\partial \rho^q} \left(e^{\imath r_j \rho \sin(\theta) - \imath n \theta} \right) \mathrm{d}\theta \right| \\ &= \left| \frac{1}{2\pi} \int_0^{2\pi} (\imath r_j \sin(\theta))^q \, e^{\imath r_j \rho \sin(\theta) - \imath n \theta} \mathrm{d}\theta \right| \\ &\leq \left(\frac{1}{2\pi} \int_0^{2\pi} \mathrm{d}\theta \right). \end{aligned}$$

In combination with Stirling's approximation [9, 5.11.3], it follows that

$$|R(\rho)| \leq \frac{1}{q!} \left(\frac{\sqrt{\pi p}}{4}\right)^q \leq \left(\frac{\sqrt{\pi p} e}{4q}\right)^q.$$

Therefore, in order to achieve error $|R(\rho)| \leq \gamma$, it suffices to set q such that

$$\gamma \geq \left(\frac{\sqrt{\pi p}e}{4q}\right)^q. \tag{4.9}$$

Setting $\sqrt{\pi p}e/4q = 1/2$ and solving for q gives

$$q = \frac{\sqrt{\pi}e\sqrt{p}}{2} \approx 2.4\sqrt{p} \quad \Longrightarrow \quad q \geq \max\{2.4\sqrt{p}, \log_2 \gamma^{-1}),$$

which is sufficient to achieve error less than γ. ∎

Remark 4.4 (improving estimates for number of radial and angular nodes). While Lemmas 4.3 and 4.2 show that the number of radial nodes q and angular nodes s are $\mathcal{O}(\sqrt{p})$, the constants in the lemmas are not optimal. For practical purposes, choosing the minimal number of nodes needed to achieve the desired error is advantageous to improve the run-time constant of the algorithm, and it is clear from the proofs how the estimates can be refined. For Lemma 4.3 we set $Q = \lceil 2.4\sqrt{p} \rceil$, and motivated by (4.9) we compute

$$\gamma^{\text{rad}}(q) = \frac{1}{\sqrt{\pi}q!}\left(\frac{\sqrt{\pi p}}{4}\right)^q$$

for $q = 1, \ldots, Q$ and choose the smallest value q^* of q such that $\gamma^{\text{rad}}(q^*) \leq \gamma$. Similarly, for Lemma 4.2, we set $S = \lceil 7.09\sqrt{p} \rceil$, and motivated by (4.6) we compute

$$\gamma^{\text{ang}}(s) = \left(\frac{\lambda_m + N_m}{s}\right)^s$$

for $s = 1, \ldots, S$ and choose the smallest value s^* of s such that $\gamma^{\text{ang}}(s) \leq \gamma$. Then, it follows that $2.4\sqrt{p}$ and $7.09\sqrt{p}$ can be replaced by q^* and s^* in the statements of Lemmas 4.3 and 4.2, respectively. This procedure improves the estimate of the required number of angular and radial nodes by a constant factor.

5. Numerical results.

Remark 5.1 (FFT bandlimit heuristic). One heuristic for setting the bandlimit is based on the fast Fourier transform (FFT). For a centered FFT on a signal of length L, the maximum frequency is $\pi^2(L/2)^2$, which corresponds to a bandlimit of $\lambda = \pi L/2$. Note that

$$\pi L/2 \approx 1.57L < 1.77L \approx \sqrt{\pi}\lfloor (L-1)/2 \rfloor/2, \tag{5.1}$$

so this FFT bandlimit heuristic does indeed produce a reasonable bandlimit below the bound (2.12) derived from Weyl's law. We use this bandlimit for our numerical experiments. The computational complexity and accuracy guarantees of the method presented in this paper hold for any bandlimit $\lambda = \mathcal{O}(L)$. However, the fact that the fast method performs interpolation in Fourier space inside a disk bounded by the maximum bandlimit provides additional motivation for this FFT-based heuristic since it will ensure that the disk will be contained within the square in frequency space used by the two-dimensional FFT.

5.1. Numerical accuracy results.

In this section, we report numerical results for the accuracy of our FDHT method compared to matrix multiplication. The implementation of the method is based on the parameters ε^{dis}, ε^{nuf}, ε^{fst}, s, and q, which

result in the error guarantees in Theorem 4.1. However, since these theoretical error bounds do not account for errors from finite precision arithmetic, the parameters used by the implementation of the algorithm are tuned slightly so that the code achieves the desired accuracy in numerical tests. In particular, when running the optimization procedure for s described in Remark 4.4, we decrease the threshold parameter γ by an additional factor of p. The motivation for this heuristic is to try to account for numerical errors. This results in a value of s slightly larger than that of Remark 4.4, but smaller than that of Lemma 4.2. In particular, this choice of s theoretically guarantees the correct complexity and accuracy. We also tune the values of $\varepsilon^{\text{dis}}, \varepsilon^{\text{nuf}}, \varepsilon^{\text{fst}}$ in Algorithms 3.1 and 3.2 for further increased performance.

Recall that $B : \mathbb{C}^m \to \mathbb{C}^p$ maps coefficients to images by

$$(B\alpha)_j = \sum_{i=1}^{m} \alpha_i \psi_i(x_j) h,$$

and its adjoint transform $B^* : \mathbb{C}^p \to \mathbb{C}^m$ maps images to coefficients by

$$(B^* f)_i = \sum_{j=1}^{p} f_j \overline{\psi_i(x_j)} h;$$

see section 1.3. By defining the $m \times p$ matrix B by

$$B_{ij} = \psi_i(x_j) h,$$

we can apply B and B^* by dense matrix multiplication to test the accuracy of our fast method. Since the size of the matrix scales like L^4 for $L \times L$ images, constructing these matrices quickly becomes prohibitive so the comparison is only given up to $L = 160$ (see Table 1), where

$$\text{err}_\alpha = \frac{\|\alpha_{\text{fast}} - \alpha_{\text{dense}}\|_{\ell^2}}{\|\alpha_{\text{dense}}\|_{\ell^2}} \quad \text{and} \quad \text{err}_f = \frac{\|f_{\text{fast}} - f_{\text{dense}}\|_{\ell^2}}{\|f_{\text{dense}}\|_{\ell^2}}$$

denote the relative errors of the coefficients and the image, respectively, where $\alpha_{\text{dense}} = B^* f$ and $f_{\text{dense}} = B\alpha$ are computed by dense matrix multiplication and α_{fast} and f_{fast} are the corresponding quantities computed using the fast algorithm of this paper.

The image used for the accuracy comparison is a tomographic projection of a 3D density map representing a bio-molecule (E. coli 70S ribosome) [35], retrieved from the online EM data bank [27].

5.2. Timing results. In this section, we plot the timing of our FDHT method for $L \times L$ images with $p = L^2$ pixels. We demonstrate that the method does indeed

TABLE 1
Relative error of fast method compared to dense matrix multiplication.

L	ε	err_α	err_f	ε	err_α	err_f
64	1.00e-04	1.92422e-05	2.10862e-05	1.00e-10	3.55320e-11	2.36873e-11
96	1.00e-04	1.82062e-05	2.52219e-05	1.00e-10	2.99849e-11	2.48166e-11
128	1.00e-04	1.90648e-05	2.41142e-05	1.00e-10	3.25650e-11	2.61890e-11
160	1.00e-04	2.00748e-05	2.49488e-05	1.00e-10	3.13903e-11	3.50455e-11
64	1.00e-07	2.03272e-08	2.98083e-08	1.00e-14	7.41374e-15	6.82660e-15
96	1.00e-07	2.28480e-08	2.58272e-08	1.00e-14	9.82890e-15	8.80843e-15
128	1.00e-07	2.69215e-08	2.27676e-08	1.00e-14	1.21146e-14	1.11909e-14
160	1.00e-07	2.47053e-08	2.51146e-08	1.00e-14	1.36735e-14	1.51430e-14

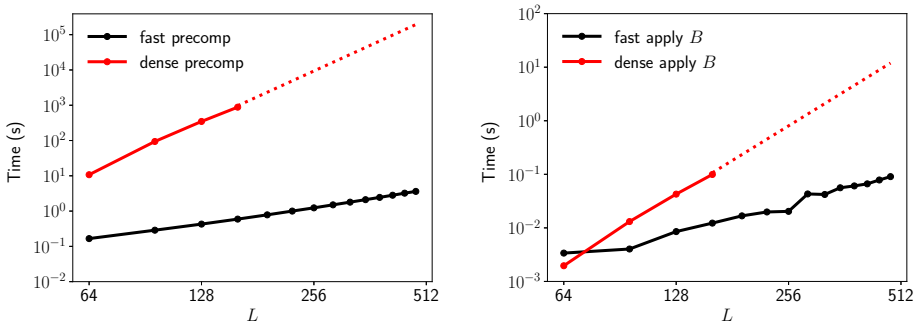

FIG. 4. *Timings of fast method versus dense method for precomputation (left) and applying B (right). The timings for the dense method for $L > 160$ are extrapolated since the memory requirements for the dense method were prohibitive.*

have complexity $\mathcal{O}(p \log p)$ and that the timings are practical. We plot the time of precomputation and the time of applying B using the fast method; for comparison, we include timings for forming and applying the dense matrix B; see Figure 4. The timings for applying B^* are similar to the timings for applying B (since the algorithm consists of applying similar transforms in the reverse order), so a separate plot was not included.

The timings were carried out on a computer with an AMD 5600X processor and 24 GB of memory. We set $\varepsilon = 10^{-7}$ for the reported timings and compare to the dense method up to $L = 160$. For $L > 160$, comparison to the dense method was prohibitively expensive. For reference, storing the dense transform matrix in double precision complex numbers for $L = 512$ would require about 640 GB of memory. The NUFFT uses the FINUFFT implementation [1, 2]. The image used for the timing results is a tomographic projection of a 3D density map representing the SARS-CoV-2 Omicron spike glycoprotein complex [22], retrieved from the online EM data bank [27].

Remark 5.2 (precomputation time negligible when transforming many images). The precomputation involves organizing Bessel function roots and creating data structures for the NUFFT and interpolation steps of the algorithm. Precomputation only needs to be performed once for a given size of image L and becomes negligible when the method is used to expand a large enough set of images (around 100 images), which is a typical use-case in, for example, applications in cryo-EM [4].

Remark 5.3 (breakdown of timing of fast algorithm). Each step of the algorithm has roughly the same magnitude. For example, for $L = 512$ and $\varepsilon = 10^{-7}$ the timings of the NUFFT, FFT, and interpolation steps of the algorithm for applying B are 0.035, 0.046, and 0.026 seconds, respectively. We note that the timing of each step is dependent on the choice of parameters. For example, sampling more points will increase the cost of the NUFFT step but decrease the cost of the interpolation step since sparser interpolation matrices can be used; decreasing ε will increase the cost of the NUFFT step.

Remark 5.4 (parallelization). The timings reported in Figure 4 are for a single-threaded CPU code. However, each step of the code is amenable to parallelization through GPU implementations. Indeed, the NUFFT step has a GPU implementation [36], and the 2D FFT and interpolation steps can also benefit from straightforward parallelization schemes.

5.3. Numerical example: Convolution and rotation. We lastly present an example illustrating the use of the steerable and fast radial convolution properties of the eigenbasis. The example is motivated by cryo-EM, wherein tomographic projection images of biological molecules in a sample are registered by electron beams; see, for example, [13] for more information. Because of aberrations within the electron-microscope and random in-plane rotations of the molecular samples, the registered image I_r does not precisely coincide with the actual projection image I_p, and the following model is used:

$$I_r(x) = c(|x|) * R_\theta\left(I_p(x)\right) + \eta, \tag{5.2}$$

where R_θ describes rotation around the origin by an angle of θ, c is a radial function termed the point-spread function, and η is additive white noise. The function $\widehat{c}$ is, in turn, known as the contrast transfer function (CTF). Examples of point spread functions are shown in Figure 5.

Notably, the regions of the frequency space where $\widehat{c}$ equals zero destroy information of $I_p(x)$. However, the fact that convolution is a diagonal transformation of the coefficients in the basis of eigenfunctions enables reconstruction of a fixed projection image I_p from a small number of registered images $I_r^{(i)}$ with different point spread functions $c_i(|x|)$, rotations R_{θ_i}, and noise $\eta^{(i)}$ for $i = 1, \ldots, t$. From Lemma 2.2, it follows that the basis coefficients $\alpha_{nk}^{(i)}$ of the registered images satisfy

$$\alpha_{nk}^{(i)} = \widehat{c}_i(\lambda_{nk}) e^{\imath n \theta_i} \alpha_{nk}^{(0)} + \eta_{nk}^{(i)} \quad \text{for } i = 1, \ldots, t, \tag{5.3}$$

where $\alpha_{nk}^{(0)}$ denote the basis coefficients of I_p. We assume that the parameters θ_i and c_i are known or estimated to a desired precision. We remark that the standard FFT can be used to solve this problem when there are no rotations.

To recover the $\alpha_{nk}^{(0)}$, we find the least-squares optimizers of (5.3). To improve the conditioning of the problem, (5.3) is thresholded to exclude the values of i for which $\widehat{c}_i(\lambda_{nk})$ has sufficiently low magnitude. We therefore estimate $\alpha_{nk}^{(0)}$ by $\alpha_{nk}^{(0)} \approx \alpha_{nk}$, with α_{nk} defined by

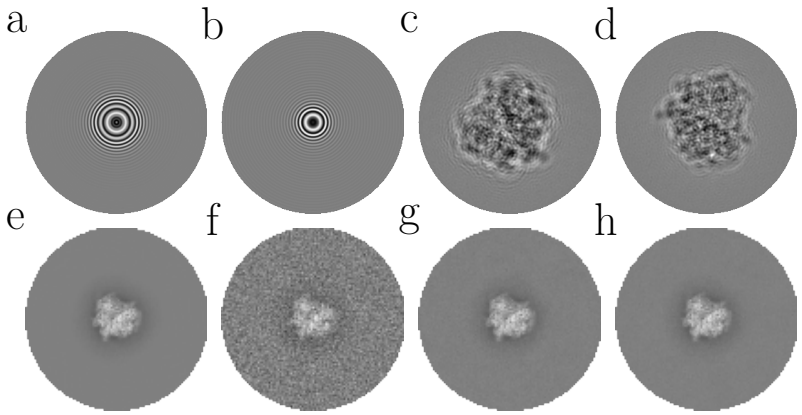

FIG. 5. *Two different point spread functions* (a)–(b)*, result of their convolution with a fixed image and subsequent rotation* (c)–(d). (e) *Projection of reference image into the eigenbasis using the fast algorithm.* (f)–(g) *Result of deconvolution algorithm using* $t = 1, 3, 5$*, respectively.*

$$\alpha_{nk} = \arg\min_{\alpha_{nk}} \sum_{(n,k)\in\mathcal{I}} \sum_{i=1}^{t} \gamma_{nk}^{(i)} \cdot \left| \alpha_{nk}^{(i)} - \widehat{c}_i(\lambda_{nk}) e^{\imath n\theta_i} \alpha_{nk} \right|^2, \tag{5.4}$$

where $\gamma_{nk}^{(i)} = 0$ if $|\widehat{c}_i(\lambda_{nk})| < \tau$, for a given threshold τ, and $\gamma_{nk}^{(i)} = 1$ otherwise. This describes a decoupled least-squares problem for each coefficient α_{nk}, which can be solved efficiently. We remark that (5.4) is a basic version of Wiener filtering [4], which we use for simplicity of exposition. The result of this procedure for different values of t and a nonzero value of the noise η is shown in Figure 5.

6. Discussion. This paper presents a fast method for expanding a set of $L \times L$ images into the basis of eigenfunctions of the Laplacian on the disk. The approach calculates the expansion coefficients from interpolation of the Fourier transform of the image on distinguished subsets of the frequency space and relies on an integral identity of the Fourier transform of the eigenfunctions. Unlike previous approaches [44], we demonstrate that our fast method is guaranteed to coincide with a dense, equivalent method up to a user-specified precision. Moreover, our method provides a natural way to compute the convolution with radial functions. Potential extensions of the presented method include extending the method to three dimensions or other domains in two dimensions.

Appendix A. Proof of Theorem 4.1. This section proves Theorem 4.1.

A.1. Proof of accuracy of Algorithm 3.1. Let $\tilde{\alpha}_i$ be the output of Algorithm 3.1, including the error from the NUFFT and fast interpolation steps. By composing the steps of the algorithm, we have

$$\tilde{\alpha}_i = c_i h \left(\sum_{k=0}^{q-1} \left(\frac{\imath^{n_i}}{s} \sum_{\ell=0}^{s-1} \left(\sum_{j=1}^{p} f_j e^{-\imath x_j \cdot \xi_{k\ell}} + \delta_{k\ell}^{\text{nuf}} \right) e^{-\imath n_i \phi_\ell} \right) u_k(\lambda_i) + \delta_i^{\text{fst}} \right),$$

where $\delta_{k\ell}^{\text{nuf}}$ and δ_i^{fst} denote the error from the NUFFT and fast interpolation, respectively. These satisfy ℓ^1-ℓ^∞ relative error bounds

$$\|\delta^{\text{nuf}}\|_{\ell^\infty} \le \varepsilon^{\text{nuf}} \sum_{j=1}^{p} |f_j e^{-\imath x_j \cdot \xi_{k\ell}}| = \varepsilon^{\text{nuf}} \|f\|_{\ell^1}, \tag{A.1}$$

and (using $\varepsilon^{\text{nuf}} \le 1$, which we can ensure holds)

$$\|\delta^{\text{fst}}\|_{\ell^\infty} \le \varepsilon^{\text{fst}} \sum_{k=0}^{q-1} \left| \frac{\imath^{n_i}}{s} \sum_{\ell=0}^{s-1} \left(\sum_{j=1}^{p} f_j e^{-\imath x_j \cdot \xi_{k\ell}} + \delta_{k\ell}^{\text{nuf}} \right) e^{-\imath n_i \phi_\ell} \right| \le 2\varepsilon^{\text{fst}} q \|f\|_{\ell^1}, \tag{A.2}$$

where ε^{nuf} and ε^{fst} are the relative error parameters for the NUFFT and fast interpolation, respectively. Let α_i denote the output of Algorithm 3.1 without the NUFFT and fast interpolation error terms, i.e.,

$$\alpha_i = c_i h \sum_{k=0}^{q-1} \frac{\imath^{n_i}}{s} \sum_{\ell=0}^{s-1} \sum_{j=1}^{p} f_j e^{-\imath x_j \cdot \xi_{k\ell}} e^{-\imath n_i \phi_\ell} u_k(\lambda_i). \tag{A.3}$$

We have

$$
\begin{aligned}
|\alpha_i - \tilde{\alpha}_i| &\le c_i h \left(\sum_{k=0}^{q-1} \frac{1}{s} \sum_{\ell=0}^{s-1} \|\delta^{\text{nuf}}\|_{\ell^\infty} |u_k(\lambda_i)| + \|\delta^{\text{fst}}\|_{\ell^\infty} \right) \\
&\le c_i h \left(\|\delta^{\text{nuf}}\|_{\ell^\infty} \sum_{k=0}^{q-1} |u_k(\lambda_i)| + \|\delta^{\text{fst}}\|_{\ell^\infty} \right) \\
&\le 2\sqrt{2} \left(\|\delta^{\text{nuf}}\|_{\ell^\infty} (2 + \frac{\pi}{2} \log q) + \|\delta^{\text{fst}}\|_{\ell^\infty} \right),
\end{aligned} \tag{A.4}
$$

where the final inequality follows from Lemma A.2 and the fact that

$$c_i h \le 2\sqrt{2},$$

which follows from the definition of h in (1.3) and Lemma A.4. Combining this equality with (A.1) and (A.2) gives

$$\|\alpha - \tilde{\alpha}\|_{\ell^\infty} \le 2\sqrt{2} \left(\varepsilon^{\text{nuf}} \left(2 + \frac{\pi}{2} \log q \right) + 2q\varepsilon^{\text{fst}} \right) \|f\|_{\ell^1}. \tag{A.5}$$

Setting

$$\varepsilon^{\text{nuf}} = \left(2\sqrt{2} \left(2 + \frac{\pi}{2} \log q \right) \right)^{-1} (\varepsilon/4) \quad \text{and} \quad \varepsilon^{\text{fst}} = (4\sqrt{2}q)^{-1} (\varepsilon/4) \tag{A.6}$$

gives

$$\|\alpha - \tilde{\alpha}\|_{\ell^\infty} \le \frac{\varepsilon}{2} \|f\|_{\ell^1}. \tag{A.7}$$

By the definition of α_i in (A.3), we have

$$
\begin{aligned}
\alpha_i &= c_i h \sum_{j=1}^{p} f_j \left(\sum_{k=0}^{q-1} \frac{\imath^{n_i}}{s} \sum_{\ell=0}^{s-1} e^{-\imath x_j \cdot \xi_{k\ell}} e^{-\imath n_i \phi_\ell} u_k(\lambda_i) \right) \\
&= c_i h \sum_{j=1}^{p} f_j \left(J_{n_i}(r_j \lambda_i) e^{-\imath n_i \theta_j} + \delta_{ij}^{\text{dis}} \right) = (B^* f)_i + c_i h \sum_{j=1}^{p} f_j \delta_{ij}^{\text{dis}},
\end{aligned} \tag{A.8}
$$

where the third equality follows from Lemma A.1 with δ_{ij}^{dis} a discretization error that is bounded by $\|\delta^{\text{dis}}\|_{\ell^\infty} \le (3 + \frac{\pi}{2} \log q) \varepsilon^{\text{dis}}$. It follows that

$$|\alpha_i - (B^* f)_i| \le c_i h \|f\|_{\ell^1} \|\delta^{\text{dis}}\|_{\ell^\infty} \le 2\sqrt{2} \left(3 + \frac{\pi}{2} \log q \right) \varepsilon^{\text{dis}} \|f\|_{\ell^1}.$$

Setting

$$\varepsilon^{\text{dis}} = \left(2\sqrt{2} \left(3 + \frac{\pi}{2} \log \left(2.4\sqrt{p} \right) \right) \right)^{-1} \frac{\varepsilon}{2} \tag{A.9}$$

and combining with (A.7) gives

$$\|\alpha - B^* f\|_{\ell^\infty} \le \varepsilon \|f\|_{\ell^1},$$

which completes the proof of the accuracy guarantees for Algorithm 3.1.

A.2. Proof of accuracy of Algorithm 3.2. Let $\tilde{f}$ be the output of Algorithm 3.2, including the approximation error from using fast interpolation and the NUFFT. By composing the steps of Algorithm 3.2 we have

$$\tilde{f}_j = \sum_{k=0}^{q-1}\sum_{\ell=0}^{s-1}\left(\sum_{n=-N_m}^{N_m}\left(\sum_{i:n_i=n} u_k(\lambda_i)c_i h\alpha_i + \delta_{nk}^{\text{fst}}\right)\frac{(-\imath)^n}{s}e^{\imath n\phi_\ell}\right)e^{-\imath x_j\cdot\xi_{k\ell}} + \delta_j^{\text{nuf}},$$

where δ_{nk}^{fst} and δ_j^{nuf} denote the error from the fast interpolation and NUFFT, respectively, which satisfy ℓ^1-ℓ^∞ relative error bounds. We have

$$\|\delta_n^{\text{fst}}\|_{\ell^\infty} \le \varepsilon^{\text{fst}} \sum_{i:n_i=n} c_i h|\alpha_i| \le 2\sqrt{2}\varepsilon^{\text{fst}} \sum_{i:n_i=n} |\alpha_i|,$$

where $\delta_n^{\text{fst}} = (\delta_{nk}^{\text{fst}})_{k=0}^{q-1}$, and

$$\begin{aligned}
\|\delta^{\text{nuf}}\|_{\ell^\infty} &\le \varepsilon^{\text{nuf}} \sum_{k=0}^{q-1}\sum_{\ell=0}^{s-1}\frac{1}{s}\left|\sum_{n=-N_m}^{N_m}\left(\sum_{i:n_i=n} u_k(\lambda_i)c_i h\alpha_i + \delta_{nk}^{\text{fst}}\right)\right| \\
&\le \varepsilon^{\text{nuf}} \sum_{i=1}^{m}\left(\sum_{k=0}^{q-1}|u_k(\lambda_i)|\right)c_i h|\alpha_i| + \varepsilon^{\text{nuf}}\sum_{k=0}^{q-1}\sum_{n=-N_m}^{N_m} 2\sqrt{2}\varepsilon^{\text{fst}}\sum_{i:n_i=n}|\alpha_i| \\
&\le \varepsilon^{\text{nuf}}\left(\left(2+\frac{\pi}{2}\log q\right)2\sqrt{2}\|\alpha\|_{\ell^1} + q2\sqrt{2}\varepsilon^{\text{fst}}\|\alpha\|_{\ell^1}\right) \\
&\le \varepsilon^{\text{nuf}} 2\sqrt{2}\left(\left(2+\frac{\pi}{2}\log q\right)+1\right)\|\alpha\|_{\ell^1},
\end{aligned} \tag{A.10}$$

where the final inequality assumes $q\varepsilon^{\text{fst}} \le 1$. Let f_j denote the output of Algorithm 3.2, ignoring the error from the fast interpolation and NUFFT, i.e.,

$$f_j = \sum_{k=0}^{q-1}\sum_{\ell=0}^{s-1}\sum_{n=-N_m}^{N_m}\sum_{i:n_i=n} u_k(\lambda_i)c_i h\alpha_i \frac{(-\imath)^n}{s}e^{\imath n\phi_\ell}e^{-\imath x_j\cdot\xi_{k\ell}}. \tag{A.11}$$

We have

$$\begin{aligned}
|f_j - \tilde{f}_j| &\le \left(\sum_{k=0}^{q-1}\sum_{\ell=0}^{s-1}\frac{1}{s}\sum_{n=-N_m}^{N_m}|\delta_{nk}^{\text{fst}}|\right) + |\delta_j^{\text{nuf}}| \\
&\le \left(\sum_{k=0}^{q-1} 2\sqrt{2}\varepsilon^{\text{fst}}\|\alpha\|_{\ell^1}\right) + \varepsilon^{\text{nuf}} 2\sqrt{2}\left(\left(2+\frac{\pi}{2}\log q\right)+1\right)\|\alpha\|_{\ell^1} \\
&\le \left(\varepsilon^{\text{fst}} 2\sqrt{2}q + \varepsilon^{\text{nuf}} 2\sqrt{2}\left(3+\frac{\pi}{2}\log q\right)\right)\|\alpha\|_{\ell^1}.
\end{aligned} \tag{A.12}$$

Setting

$$\varepsilon^{\text{nuf}} = \left(2\sqrt{2}\left(3+\frac{\pi}{2}\log q\right)\right)^{-1}(\varepsilon/4) \quad \text{and} \quad \varepsilon^{\text{fst}} = (2\sqrt{2}q)^{-1}(\varepsilon/4) \tag{A.13}$$

gives

$$\|f - \tilde{f}\|_{\ell^\infty} \le \frac{\varepsilon}{2}\|\alpha\|_{\ell^1}.$$

By the definition of f_j in (A.11), we have

$$
\begin{aligned}
f_j &= \sum_{k=0}^{q-1}\sum_{\ell=0}^{s-1}\sum_{i=1}^{m} u_k(\lambda_i)c_i h\alpha_i \frac{(-\imath)^n}{s} e^{\imath n\phi_\ell} e^{-\imath x_j\cdot\xi_{k\ell}} \\
&= \sum_{i=1}^{m} c_i h\alpha_i \left(\sum_{k=0}^{q-1} \frac{(-\imath)^n}{s} \sum_{\ell=0}^{s-1} u_k(\lambda_i) e^{\imath n\phi_\ell} e^{-\imath x_j\cdot\xi_{k\ell}} \right) \qquad \text{(A.14)} \\
&= \sum_{i=1}^{m} c_i h\alpha_i \left(J_{n_i}(r_j\lambda_i) e^{\imath n_i\theta_j} + \bar{\delta}_{ij}^{\mathrm{dis}} \right) = (Bf)_j + \sum_{i=1}^{m} c_i h\alpha_i \bar{\delta}_{ij}^{\mathrm{dis}},
\end{aligned}
$$

where the third equality follows from Lemma A.1 with $\bar{\delta}_{ij}^{\mathrm{dis}}$ a discretization error that satisfies $\|\bar{\delta}^{\mathrm{dis}}\|_{\ell^\infty} \le (3+\frac{\pi}{2}\log q)\varepsilon^{\mathrm{dis}}$. It follows that

$$
\|f - B\alpha\|_{\ell^\infty} \le 2\sqrt{2}\left(3\frac{\pi}{2}\log q\right)\varepsilon^{\mathrm{dis}}\|\alpha\|_{\ell^\infty}.
$$

Setting

$$
\varepsilon^{\mathrm{dis}} = \left(2\sqrt{2}\left(3+\frac{\pi}{2}\log\left(2.4\sqrt{p}\right)\right)\right)^{-1}\frac{\varepsilon}{2} \qquad \text{(A.15)}
$$

and combining with (A.7) gives

$$
\|f - B\alpha\|_{\ell^\infty} \le \varepsilon\|\alpha\|_{\ell^1},
$$

which completes the proof of the accuracy guarantees for Algorithm 3.2.

A.3. Proof of computational complexity of Algorithms 3.1 and 3.2.

A.3.1. Computational complexity of NUFFT. Both Algorithms 3.1 and 3.2 use the same number of source points and target points and have asymptotically similar error parameters, and thus have the same computational complexity. In both cases, the number of source points is p, the number of target points is $sq = \mathcal{O}(p)$ (see the definition of s and q in Lemmas 4.2 and 4.3), and the error parameter $\varepsilon^{\mathrm{nuf}} = \mathcal{O}(\varepsilon/\log q)$; see (A.6) and (A.13). It follows that the computational complexity is $\mathcal{O}(p\log p + p\,|\log\varepsilon - \log\log q|^2)$ (see (3.1) or [1, 2]), which simplifies to $\mathcal{O}(p\log p + p\,|\log\varepsilon|^2)$ operations.

A.3.2. Computational complexity of FFT. Algorithms 3.1 and 3.2 use an FFT and inverse FFT (which both have the same computational complexity) on a similar amount of data. In particular, they perform $\mathcal{O}(\sqrt{p})$ applications of the FFT of size $\mathcal{O}(\sqrt{p})$. Therefore, the computational complexities are $\mathcal{O}(p\log p)$.

A.3.3. Computational complexity of fast interpolation. There are a number of ways to perform fast polynomial interpolation; see Remark 3.1. For consistency with the rest of the paper, assume that fast interpolation is performed using the NUFFT, whose computational complexity in dimension d is stated in (3.1).

Recall that $N_m = \max\{n_j \in \mathbb{Z} : j \in \{1,\dots,m\}\}$ and $K_n = \max\{k \in \mathbb{Z}_{>0} : \lambda_{nk} \le \lambda \text{ for some } n \in \mathbb{Z}\}$. By (4.7) we have $N_m \le \sqrt{\pi p}$. Fix $n \in \{-N_m,\dots,N_m\}$; we need to compute a polynomial interpolation from $q = \mathcal{O}(\sqrt{p})$ source points (Chebyshev nodes) to K_n target nodes. The computational complexity of each interpolation problem to ℓ^1-ℓ^∞ relative error δ is $\mathcal{O}(\sqrt{p}\log p + K_n|\log\delta|)$. Summing over the $2N_m+1 = \mathcal{O}(\sqrt{p})$ interpolation problems gives a total complexity of $\mathcal{O}(p\log p + p|\log\delta|)$, where we used the fact that $\sum_{n=-N_m}^{N_m} K_n = m = \mathcal{O}(p)$. It follows from (A.6) and (A.13) that the computational complexities are $\mathcal{O}(p\log p + p|\log\varepsilon|)$.

A.3.4. Summary. Since all of the steps are $\mathcal{O}(p \log p + p|\log \varepsilon|^2)$, the proof is complete.

A.4. Technical lemmas. We first state and prove a lemma that combines Lemma 4.2 and Lemma 4.3.

LEMMA A.1. *Let s and q be defined by Lemmas* 4.2 *and* 4.3 *with accuracy parameter $\gamma > 0$. Then*

$$\left| \sum_{k=0}^{q-1} \frac{\imath^{n_i}}{s} \sum_{\ell=0}^{s-1} e^{-\imath x_j \cdot \xi_{k\ell}} e^{-\imath n_i \phi_\ell} u_k(\lambda_i) - J_n(r_j \lambda_i) e^{-\imath n_i \theta_j} \right| \leq \left(3 + \frac{\pi}{2} \log q\right) \gamma$$

for $i \in \{1, \ldots, m\}$ and $j \in \{1, \ldots, p\}$.

Proof. We can write

$$\begin{aligned} \sum_{k=0}^{q-1} \frac{\imath^{n_i}}{s} \sum_{\ell=0}^{s-1} e^{-\imath x_j \cdot \xi_{k\ell}} e^{-\imath n_i \phi_\ell} u_k(\lambda_i) &= \sum_{k=0}^{q-1} \left(J_n(r_j t_k) e^{-\imath n_i \theta_j} + \delta_{kij}^{\text{ang}} \right) u_k(\lambda_i) \\ &= J_n(r_j \lambda_i) e^{-\imath n_i \theta_j} + \delta_{ij}^{\text{rad}} + \sum_{k=0}^{q-1} \delta_{kij}^{\text{ang}} u_k(\lambda_i), \end{aligned} \tag{A.16}$$

where $\delta_{kij}^{\text{ang}}$ and δ_{ij}^{rad} are the errors from discretizing the angles and using interpolation in the radial direction, respectively. By Lemmas 4.2 and 4.3 it follows that the error satisfies

$$\left| \delta_{ij}^{\text{rad}} + \sum_{k=0}^{q-1} \delta_{kij}^{\text{ang}} u_k(\lambda_i) \right| \leq \gamma + \gamma \left(2 + \frac{\pi}{2} \log q\right),$$

which completes the proof. ∎

We will use the following property of Chebyshev interpolation polynomials; see [34, eq. 11].

LEMMA A.2. *Let t_k be Chebyshev nodes of the first kind defined in* (3.3) *for the interval $[\lambda_1, \lambda_m]$. Then,*

$$\sum_{k=0}^{q-1} |u_k(t)| \leq 2 + \frac{2}{\pi} \log q, \quad \textit{where} \quad u_k(t) = \frac{\prod_{\ell \neq k} (t - t_\ell)}{\prod_{\ell \neq k} (t_k - t_\ell)},$$

for all $t \in [\lambda_1, \lambda_m]$.

We will also require a classical result on discretization errors for integrals of smooth periodic functions.

LEMMA A.3. *Suppose that $g : [0, 2\pi] \to \mathbb{C}$ is a smooth periodic function on the torus $[0, 2\pi]$ where 0 and 2π are identified. Then*

$$\left| \int_0^{2\pi} g(\phi) d\phi - \frac{2\pi}{s} \sum_{\ell=0}^{s-1} g(\phi_\ell) \right| < 4 \frac{\|g^{(s)}\|_{L^1}}{s^s} \tag{A.17}$$

for all $s \geq 2$, where $\phi_\ell = 2\pi\ell/s$, where $g^{(s)}(\phi)$ denotes the sth derivative of $g(\phi)$ with respect to ϕ.

See [24, Theorem 1.1] for a proof. Lastly, we prove an upper bound on the normalization constants c_{nk}.

LEMMA A.4. *If $\lambda_{nk} \le \sqrt{\pi p}$, then the constants c_{nk} satisfy $|c_{nk}| \le \sqrt{2p}$.*

Proof of Lemma A.4. We start with an alternate equivalent definition to (2.2):

$$c_{nk} = \frac{1}{|\pi^{1/2} J_n'(\lambda_{nk})|} \quad \text{for} \quad (n,k) \in \mathbb{Z} \times \mathbb{Z}_{>0}; \tag{A.18}$$

see [9, eq. 10.6.3, eq. 10.22.37]. By [9, eq. 10.18.4, eq. 10.18.6],

$$J_n(x) = M_n(x)\cos(\theta_n(x)),$$

where $M_n(x)^2 = J_n(x)^2 + Y_n(x)^2$ is a magnitude function, Y_n is the nth order Bessel function of the second kind, and $\theta_n(x)$ is a phase function. Taking the derivative gives

$$J_n'(x) = M_n'(x)\cos(\theta_n(x)) - M_n(x)\sin(\theta_n(x))\theta_n'(x).$$

By [20, eq. 8.479], we have

$$\frac{\pi}{2\sqrt{x^2 - n^2}} \ge M_n(x)^2 \ge \frac{\pi}{2x}. \tag{A.19}$$

In particular, the magnitude function $M_n(x)$ does not vanish, so at a root λ_{nk} of J_n, we must have $\theta_n(\lambda_{nk}) = \frac{\pi}{2} + \pi\ell$ for $\ell \in \mathbb{Z}$. It follows that

$$J_n'(\lambda_{nk})^2 = M_n(\lambda_{nk})^2 \theta_n'(\lambda_{nk})^2.$$

Using [9, eq. 10.18.8] and (A.19) gives

$$J_n'(\lambda_{nk})^2 = \left(\frac{2}{\pi\lambda_{nk}}\right)^2 M_n(\lambda_{nk})^{-2} \ge \left(\frac{2}{\pi\lambda_{nk}}\right)^2 \frac{2\sqrt{\lambda_{nk}^2 - n^2}}{\pi}.$$

By [12, eq. 1.6] we have $\lambda_{nk} > n + k\pi - \pi/2 + 1/2 > n + 2$ for $(n,k) \in \mathbb{Z}_{\ge 0} \times \mathbb{Z}_{>0}$, which implies $\sqrt{\lambda_{nk}^2 - n^2} \ge 2$ (this bound can be refined but is sufficient for the purpose of proving this lemma). Using this inequality together with the fact that $2(2/\pi)^3 \ge 1/2$ gives

$$c_{nk} = \frac{1}{\pi^{1/2}|J_n'(\lambda_{nk})|} \le \frac{2^{1/2}\lambda_{nk}}{\pi^{1/2}} \le \sqrt{2p},$$

where the final inequality follows from the assumption $\lambda_{nk} \le \sqrt{\pi p}$. ∎

Acknowledgments. The authors would like to thank Joakim Andén, Yunpeng Shi, and Gregory Chirikjian for their helpful comments on a draft of this paper. We also thank the two anonymous reviewers for their comments, which improved the exposition of the manuscript.

REFERENCES

[1] A. H. BARNETT, *Aliasing error of the* $\exp(\beta\sqrt{1-z^2})$ *kernel in the nonuniform fast Fourier transform*, Appl. Comput. Harmon. Anal., 51 (2021), pp. 1–16.

[2] A. H. BARNETT, J. MAGLAND, AND L. AF KLINTEBERG, *A parallel nonuniform fast Fourier transform library based on an "Exponential of semicircle" kernel*, SIAM J. Sci. Comput., 41 (2019), pp. C479–C504, https://doi.org/10.1137/18M120885X.

[3] J.-P. BERRUT AND L. N. TREFETHEN, *Barycentric Lagrange interpolation*, SIAM Rev., 46 (2004), pp. 501–517, https://doi.org/10.1137/S0036144502417715.

[4] T. Bhamre, T. Zhang, and A. Singer, *Denoising and covariance estimation of single particle cryo-EM images*, J. Struct. Biol., 195 (2016), pp. 72–81.
[5] X. Cheng, Q. Qiu, R. Calderbank, and G. Sapiro, *RotDCF: Decomposition of Convolutional Filters for Rotation-Equivariant Deep Networks*, preprint, arXiv:1805.06846, 2018.
[6] Y. Cheng, N. Grigorieff, P. A. Penczek, and T. Walz, *A primer to single-particle cryo-electron microscopy*, Cell, 161 (2015), pp. 438–449.
[7] T. S. Cohen and M. Welling, *Steerable CNNs*, preprint, arXiv:1612.08498, 2016.
[8] Y. Colin de Verdière, *On the remainder in the Weyl formula for the Euclidean disk*, Séminaire de Théorie Spectrale et Géométrie, 29 (2010), pp. 1–13.
[9] 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., *NIST Digital Library of Mathematical Functions*, Release 1.1.5 of 2022-03-15, http://dlmf.nist.gov/.
[10] A. Dutt, M. Gu, and V. Rokhlin, *Fast algorithms for polynomial interpolation, integration, and differentiation*, SIAM J. Numer. Anal., 33 (1996), pp. 1689–1711, https://doi.org/10.1137/0733082.
[11] A. Dutt and V. Rokhlin, *Fast Fourier transforms for nonequispaced data*, SIAM J. Sci. Comput., 14 (1993), pp. 1368–1393, https://doi.org/10.1137/0914081.
[12] A. Elbert, *Some recent results on the zeros of Bessel functions and orthogonal polynomials*, J. Comput. Appl. Math., 133 (2001), pp. 65–83.
[13] J. Frank, *Three-Dimensional Electron Microscopy of Macromolecular Assemblies: Visualization of Biological Molecules in Their Native State*, Oxford University Press, Oxford, UK, 2006.
[14] C. F. Gauss, *De nexu inter multitudinem classium*, in Quas formae binariae secundi gradus distribuuntur, earumque determinantem, Cambridge Library Collection–Mathematics 2, Cambridge University Press, Cambridge, UK, 2011, pp. 269–291, https://doi.org/10.1017/CBO9781139058230.012.
[15] A. Ghaani Farashahi and G. S. Chirikjian, *Fourier-Zernike series of convolutions on disks*, Mathematics, 6 (2018), 290.
[16] A. Ghaani Farashahi and G. S. Chirikjian, *Discrete Spectra of Convolutions on Disks Using Sturm-Liouville Theory*, preprint, arXiv:1901.05001, 2019.
[17] A. Ghaani Farashahi and G. S. Chirikjian, *Fourier-Zernike series of compactly supported convolutions on SE(2)*, J. Approx. Theory, 271 (2021), 105621.
[18] A. Ghaani Farashahi and G. S. Chirikjian, *Fourier-Bessel series of compactly supported convolutions on disks*, Anal. Appl., 20 (2022), pp. 171–192.
[19] Z. Gimbutas, N. F. Marshall, and V. Rokhlin, *A fast simple algorithm for computing the potential of charges on a line*, Appl. Comput. Harmon. Anal., 49 (2020), pp. 815–830.
[20] I. S. Gradshteyn and I. M. Ryzhik, *Table of Integrals, Series, and Products*, Academic Press, New York, 2014.
[21] L. Greengard and J.-Y. Lee, *Accelerating the nonuniform fast Fourier transform*, SIAM Rev., 46 (2004), pp. 443–454, https://doi.org/10.1137/S003614450343200X.
[22] H. Guo, Y. Gao, T. Li, T. Li, Y. Lu, L. Zheng, Y. Liu, T. Yang, F. Luo, S. Song, et al., *Structures of Omicron spike complexes and implications for neutralizing antibody development*, Cell Rep., 39 (2022), 110770.
[23] G. H. Hardy and E. Landau, *The lattice points of a circle*, Proc. R. Soc. Lond. Ser. A, Contain. Pap. Math. Phys. Character, 105 (1924), pp. 244–258.
[24] A. Kurganov and J. Rauch, *The order of accuracy of quadrature formulae for periodic functions*, in Advances in Phase Space Analysis of Partial Differential Equations, Springer, New York, 2009, pp. 155–159.
[25] B. Landa and Y. Shkolnisky, *Approximation scheme for essentially bandlimited and space-concentrated functions on a disk*, Appl. Comput. Harmon. Anal., 43 (2017), pp. 381–403.
[26] B. Landa and Y. Shkolnisky, *Steerable principal components for space-frequency localized images*, SIAM J. Imaging Sci., 10 (2017), pp. 508–534, https://doi.org/10.1137/16M1085334.
[27] C. L. Lawson, A. Patwardhan, M. L. Baker, C. Hryc, E. S. Garcia, B. P. Hudson, I. Lagerstedt, S. J. Ludtke, G. Pintilie, R. Sala, et al., *EMDataBank unified data resource for 3DEM*, Nucleic Acids Res., 44 (2016), pp. D396–D403.
[28] J.-Y. Lee and L. Greengard, *The type 3 nonuniform FFT and its applications*, J. Comput. Phys., 206 (2005), pp. 1–5.
[29] E. Nogales and S. H. W. Scheres, *Cryo-EM: A unique tool for the visualization of macromolecular complexity*, Mol. Cell, 58 (2015), pp. 677–689.

[30] M. O'Neil, F. Woolfe, and V. Rokhlin, *An algorithm for the rapid evaluation of special function transforms*, Appl. Comput. Harmon. Anal., 28 (2010), pp. 203–226, https://doi.org/10.1016/j.acha.2009.08.005.
[31] G. A. Papakostas, Y. S. Boutalis, D. A. Karras, and B. G. Mertzios, *A new class of Zernike moments for computer vision applications*, Inform. Sci., 177 (2007), pp. 2802–2819.
[32] G. Plonka, D. Potts, G. Steidl, and M. Tasche, *Numerical Fourier Analysis*, Springer, New York, 2018.
[33] A. Rangan, M. Spivak, J. Andén, and A. Barnett, *Factorization of the translation kernel for fast rigid image alignment*, Inverse Problems, 36 (2020), 024001.
[34] V. Rokhlin, *A fast algorithm for the discrete Laplace transformation*, J. Complexity, 4 (1988), pp. 12–32.
[35] T. R. Shaikh, H. Gao, W. T. Baxter, F. J. Asturias, N. Boisset, A. Leith, and J. Frank, *SPIDER image processing for single-particle reconstruction of biological macromolecules from electron micrographs*, Nat. Protoc., 3 (2008), pp. 1941–1974.
[36] Y.-h. Shih, G. Wright, J. Andén, J. Blaschke, and A. H. Barnett, *cuFINUFFT: A load-balanced GPU library for general-purpose nonuniform FFTs*, in 2021 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), IEEE, 2021, pp. 688–697.
[37] D. Slepian and H. O. Pollak, *Prolate spheroidal wave functions, Fourier analysis and uncertainty*—I, Bell Syst. Tech. J., 40 (1961), pp. 43–63.
[38] N. M. Temme, *Special Functions: An Introduction to the Classical Functions of Mathematical Physics*, John Wiley & Sons, New York, 1996.
[39] A. Townsend, *A fast analysis-based discrete Hankel transform using asymptotic expansions*, SIAM J. Numer. Anal., 53 (2015), pp. 1897–1917, https://doi.org/10.1137/151003106.
[40] J. G. Van der Corput, *Neue zahlentheoretische Abschätzungen*, Math. Ann., 89 (1923), pp. 215–254.
[41] R. H. Wade, *A brief look at imaging and contrast transfer*, Ultramicroscopy, 46 (1992), pp. 145–156.
[42] M. Weiler, F. A. Hamprecht, and M. Storath, *Learning steerable filters for rotation equivariant CNNs*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 849–858.
[43] von F. Zernike, *Beugungstheorie des schneidenver-fahrens und seiner verbesserten form, der phasenkontrastmethode*, Physica, 1 (1934), pp. 689–704.
[44] Z. Zhao, Y. Shkolnisky, and A. Singer, *Fast steerable principal component analysis*, IEEE Trans. Comput. Imaging, 2 (2016), pp. 1–12.
[45] Z. Zhao and A. Singer, *Rotationally invariant image representation for viewing direction classification in cryo-EM*, J. Struct. Biol., 186 (2014), pp. 153–166.
[46] R. Zhou and N. Grisouard, *Spectral Solver for Cauchy Problems in Polar Coordinates Using Discrete Hankel Transforms*, preprint, arXiv:2210.09736, 2022.