

An application of the kernel-independent FMM
using Fourier representations of integral equations

Victor Churchill

December 14, 2015

1 Problem formulation

The original KIFMM paper by Ying, Biros, and Zorin, explored three data sets for the 3D case: densities distributed on the unit sphere, densities distributed uniformly on the unit cube, and densities distributed at the eight corners of the unit cube.

1.1 Contribution of this project

This project examines the case of a broader category of non-uniform distributions, surfaces of revolution. In particular, we look at surfaces of revolution that are rotationally symmetric with respect to the azimuthal angle, or axisymmetric as in Figure 1. While the original 3D KIFMM may be able to handle surfaces of revolution, [4] explains that the implementation is more difficult than the 2D case. To work around that issue, this project proposes implementing the technique described in [5] to approximate solutions of 3D boundary integral equations using 2D Fourier modes.

2 Background

Here we review key concepts of the KIFMM and the Fourier representation of boundary integral equations.

2.1 KIFMM

The kernel-independent fast multipole method uses a continuous distribution of an equivalent density on a surface enclosing a box in the hierarchical tree to represent the potential generated by sources in that box, rather than using analytic multipole expansions as in the original FMM. The upward and downward formulation of these equivalent densities and their translation is explained in detail in §3.2 and shown in Figures 3, 4, 5, and 6. This allows us to construct an efficient FMM that only requires kernel evaluations. The KIFMM is also relatively easy to implement, since it

applies to an arbitrary kernel that is the fundamental solution of a second order elliptic PDE. To change the kernel in the original FMM, one would need to develop analytic multipole expansions for that kernel that may be difficult to produce.

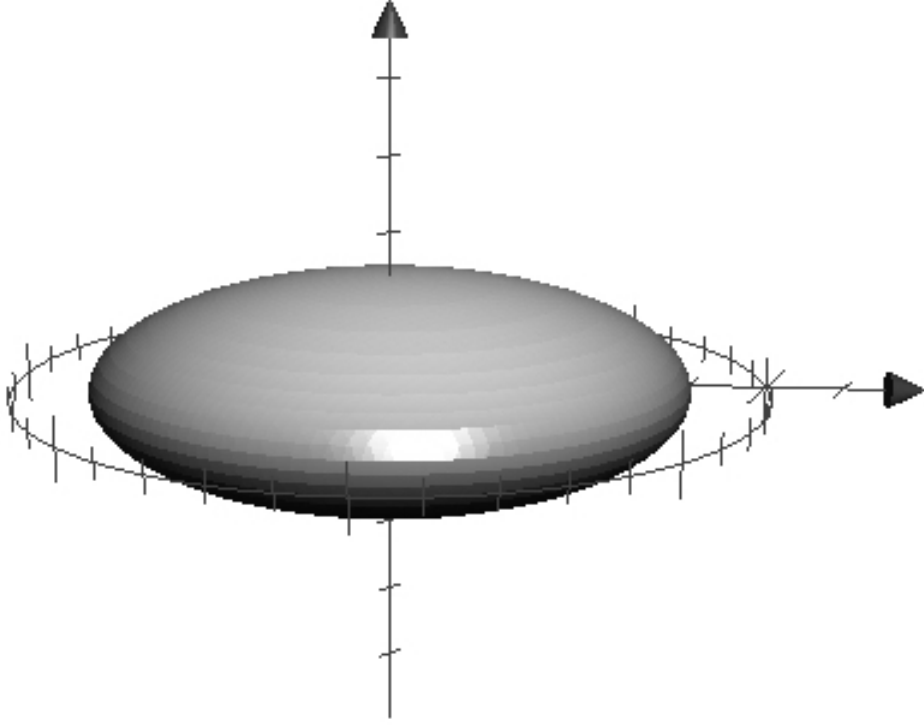


Figure 1: An axisymmetric surface Γ in 3D.

2.2 Fourier representation of integral equations

Our strategy to apply the KIFMM to axisymmetric surfaces of revolution is grounded in the idea that it is easier to solve boundary integral equations defined on curves in \mathbb{R}^2 than those defined on surfaces in \mathbb{R}^3 . This section is based on results in [5].

Consider the Fredholm integral equation of the first kind defined on the axisymmetric surface Γ

in 3D:

$$\int_{\Gamma} k(\mathbf{x}, \mathbf{y}) \phi(\mathbf{y}) d\mathbf{y} = q(\mathbf{x}) \quad \mathbf{x} \in \Gamma \quad (1)$$

where k is a kernel function, ϕ is an unknown density, and q is a potential. This is exactly the same type of integral we see in the 3D KIFMM for example in the $S2M$ step in [4].

The surface Γ is obtained by rotating a curve γ about the z axis. γ is called the generating curve, shown in Figure 2. In particular, $\Gamma = \gamma \times \mathbb{T}$ where \mathbb{T} is the one-dimensional torus (circle) parametrized by $\theta \in (-\pi, \pi]$. Since $k(\mathbf{x}, \mathbf{y})$ is axisymmetric, we have that k is a function only of the difference between θ and θ' :

$$k(\mathbf{x}, \mathbf{y}) = k(\theta - \theta', r, z, r', z') \quad (2)$$

where $\mathbf{x} = (r, z, \theta)$ and $\mathbf{y} = (r', z', \theta')$ in 3D cylindrical coordinates.

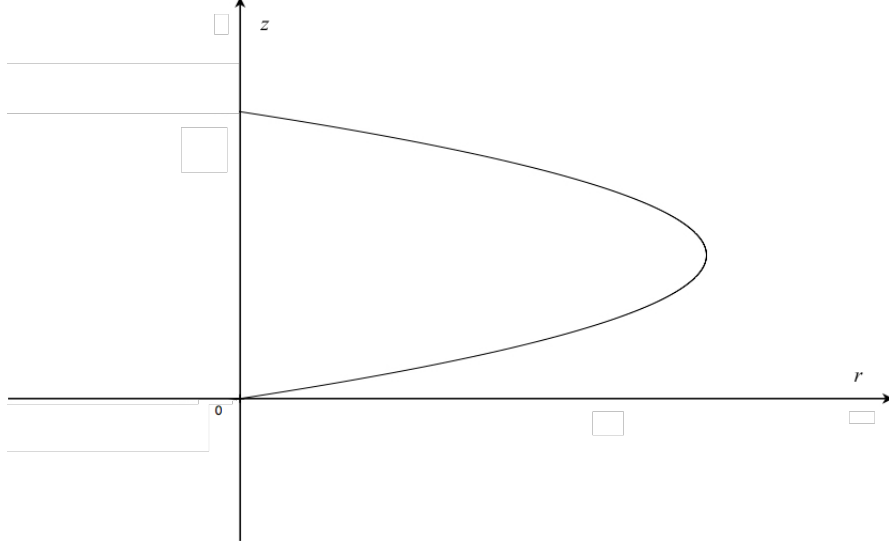


Figure 2: The generating curve γ in 2D, of an axisymmetric surface Γ in 3D.

Due to these convenient circumstances, we can restate (1) as a sequence of integral equations

defined on the generating curve by performing a Fourier transform. If ϕ_n , q_n , and k_n are the Fourier modes of k , ϕ , and q , then (1) becomes:

$$\sqrt{2\pi} \int_{\gamma} k_n(r, z, r', z') \phi_n(r', z') r' dl(r', z') = q_n(r, z) \quad (r, z) \in \gamma, n \in \mathbb{Z} \quad (3)$$

where we have:

$$q_n(r, z) = \int_{\mathbb{T}} \frac{e^{-in\theta}}{\sqrt{2\pi}} q(r, z, \theta) d\theta \quad q(\mathbf{x}) = \sum_{n \in \mathbb{Z}} \frac{e^{in\theta}}{\sqrt{2\pi}} q_n(r, z) \quad (4)$$

$$\phi_n(r, z) = \int_{\mathbb{T}} \frac{e^{-in\theta}}{\sqrt{2\pi}} \phi(r, z, \theta) d\theta \quad \phi(\mathbf{x}) = \sum_{n \in \mathbb{Z}} \frac{e^{in\theta}}{\sqrt{2\pi}} \phi_n(r, z) \quad (5)$$

$$k_n(r, z, r', z') = \int_{\mathbb{T}} \frac{e^{-in\theta}}{\sqrt{2\pi}} k(r, z, r', z', \theta) d\theta \quad k(\mathbf{x}, \mathbf{y}) = \sum_{n \in \mathbb{Z}} \frac{e^{in\theta}}{\sqrt{2\pi}} k_n(r, z, r', z') \quad (6)$$

In §3.2, we'll apply this same transformation repeatedly to 3D integral equations defined in the KIFMM, approximating with a series of 2D integral equations that are easier to solve. For convenience, we can rewrite (3) as:

$$K_n \phi_n = q_n \quad (7)$$

If K_n is a continuously invertible operator, we have:

$$\phi_n = K_n^{-1} q_n \quad (8)$$

When we implement this strategy in combination with the KIFMM, we'll use Tikhonov regularization to stably solve (8). Plugging in to (6) gives the solution of the initial 3D surface integral

equation (1):

$$\phi(r, z, \theta) = \sum_{n \in \mathbb{Z}} \frac{e^{in\theta}}{\sqrt{2\pi}} [K_n^{-1} q_n](r, z) \quad (9)$$

In practice, we choose a truncation parameter, N , such that $\|q - \sum_{n=-N}^N \frac{e^{in\theta}}{\sqrt{2\pi}} q_n\| \leq \epsilon$:

$$\phi_{approx} = \sum_{n=-N}^N \frac{e^{in\theta}}{\sqrt{2\pi}} K_n^{-1} q_n \quad (10)$$

Note that in our KIFMM implementation, the real potential q_n for each Fourier mode, is actually only used for computing error of the equivalent density.

3 Application of KIFMM

In this section we use the Fourier representation of surface BIEs described in §2.2 together with the 2D KIFMM. Notation in this section for the surfaces follows [4].

3.1 Single-layer 3D Laplace kernel

For now, we apply the algorithm considering only the single-layer 3D Laplace kernel. This choice has been made because its Fourier modes can be solved for analytically. For many other kernels, that is not that case, and we would need to approximate the modes using a discretization. This is left for future work.

We find the Fourier modes for this kernel below as in [5]. For $\mathbf{x} = (r, z, \theta)$ and $\mathbf{y} = (r', z', \theta')$:

$$k(\mathbf{x}, \mathbf{y}) = \frac{1}{4\pi|\mathbf{x} - \mathbf{y}|} \quad (11)$$

$$= \frac{1}{4\pi\sqrt{r^2 + r'^2 - 2rr'\cos(\theta - \theta') + (z - z')^2}} \quad (12)$$

$$= \sum_{n \in \mathbb{Z}} \frac{e^{in\theta}}{\sqrt{2\pi}} k_n(r, z, r', z') \quad (13)$$

$$\text{where } k_n(r, z, r', z') = \frac{1}{\sqrt{8\pi^3 rr'}} Q_{n-\frac{1}{2}} \left(\frac{r^2 + (r')^2 + (z - z')^2}{2rr'} \right) \quad (14)$$

and $Q_{n-\frac{1}{2}}$ is the half-integer order Legendre function of the second kind.

Notice that k_n does not depend only on the difference between each coordinate, e.g. $(r - r')$. This has implications for the translation operators we construct for the KIFMM which is discussed in §3.2.1.

3.2 Full algorithm

Now that we have our kernel k_n in 2D, we proceed through the regular KIFMM algorithm for the sources on the 2D generating curve γ of the surface Γ **for each** $n \in [-N, -N+1, \dots, N]$ where N is the truncation parameter chosen earlier in (10). The KIFMM algorithm is very similar to the FMM algorithm described in [1], apart from how the equivalent densities are represented, and how the translation operators are computed. As mentioned earlier, rather than by multipole expansions, the potential due to sources in a box is matched to an equivalent density at discretization points on a surface enclosing the box. In 2D, this is a circle with radius prescribed in [4]. To find these equivalent densities, we will need to discretize several integral operators on different surfaces, which is explained in §3.2.2.

First, for each mode n find the upward equivalent density for each leaf box. In the following equations, $\mathbf{x} = (r, z)$ and $\mathbf{y} = (r', z')$. Note the r' under each integral, recall that this is actually

part of the integral operator K_n as in (3).

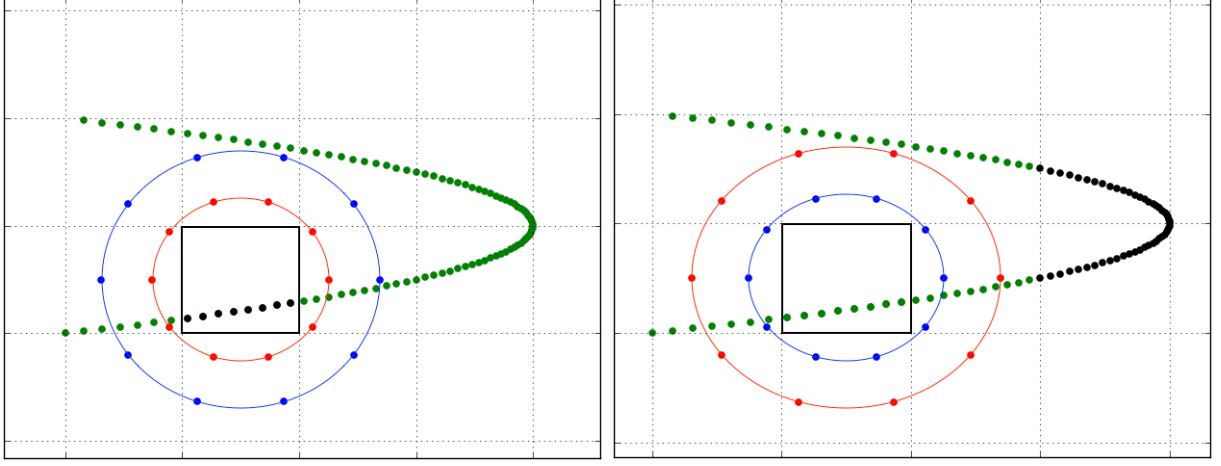


Figure 3: Left: The upward check (blue) and equivalent (red) surfaces of a box used to compute the upward equivalent density due to source densities (black) in the box. Right: The downward check (blue) and equivalent (red) surfaces of a box used to compute the downward equivalent density due to sources densities (black) in the far field on the box.

Recall that in the 3D KIFMM, we needed to solve 3D integral equations like:

$$\text{S2M: } \int_{\mathbf{y}^{B,u}} k(\mathbf{x}, \mathbf{y}) \phi^{B,u}(\mathbf{y}) d\mathbf{y} = \sum_{i \in I_s^B} k(\mathbf{x}, \mathbf{y}_i) \phi_i \text{ for all } \mathbf{x} \in \mathbf{x}^{B,u} \quad (15)$$

The reason we can write (15) as a series of equations like (16) (in 2D) is because of the Fourier transformation performed. If this is still confusing, equation (15) is to (16) as equation (1) is to (3). Similar to the **S2M** step in the FMM, solving the following equation for $\phi^{B,u}$ gives the upward equivalent density for a box B in the KIFMM:

$$\text{S2M: } \int_{\mathbf{y}^{B,u}} k_n(\mathbf{x}, \mathbf{y}) \phi_n^{B,u}(\mathbf{y}) r' d\mathbf{y} = \sum_{i \in I_s^B} k_n(\mathbf{x}, \mathbf{y}_i) \phi_i r'_i \text{ for all } \mathbf{x} \in \mathbf{x}^{B,u} \quad (16)$$

$$\text{Discretized S2M: } M_n \phi_n^{B,u} = q_n^{B,u} \quad (17)$$

where in (17), M_n is the matrix of kernel evaluations of pairs of discretization points on the upward

check surface and upward equivalent surface of a box B , $q_n^{B,u}$ is the upward check potential, and the densities at the actual source points in the box are ϕ_i . This is similar to (7), and this process is illustrated in Figure 3. Once the upward equivalent density is computed for each leaf box, we use translation operators to find the upward and downward equivalent densities for every other box.

3.2.1 Translation Operators

Translation operators limit the number of equivalent densities we need to compute directly by translating upward equivalent densities of the leaf boxes to upward and downward equivalent densities of all other boxes. In the KIFMM in particular, the translation operators in their discretized form are matrices that translate an equivalent density from one box to that of another. They are a pre-computation, as they are constant regardless of the source distribution.

In the previous step, we computed the upward equivalent density for each leaf box. The **M2M** operator translates the upward equivalent density from a leaf box A to the upward equivalent density of its parent box B . Solving the following equation for $\phi_n^{B,u}$ gives the M2M operator.

$$\text{M2M: } \int_{\mathbf{y}^{B,u}} k_n(\mathbf{x}, \mathbf{y}) \phi_n^{B,u}(\mathbf{y}) r' d\mathbf{y} = \int_{\mathbf{y}^{A,u}} k_n(\mathbf{x}, \mathbf{y}) \phi_n^{A,u}(\mathbf{y}) r' d\mathbf{y} \text{ for all } \mathbf{x} \in \mathbf{x}^{B,u} \quad (18)$$

$$\text{Discretized M2M: } M_n^B \phi_n^{B,u} = M_n^A \phi_n^{A,u} \quad (19)$$

where M_n^A is the matrix of kernel evaluations of pairs of discretization points on the upward check surface of box B and the upward equivalent surface of box A , and M_n^B is the matrix of kernel evaluations of pairs of discretization points on the upward check surface of box B and the upward equivalent surface of box B . Figure 4 gives a graphical representation of this step.

After repeating that step, every box now has an upward equivalent density. The **M2L** operator translates the upward equivalent density from a non-leaf box A to the downward equivalent density

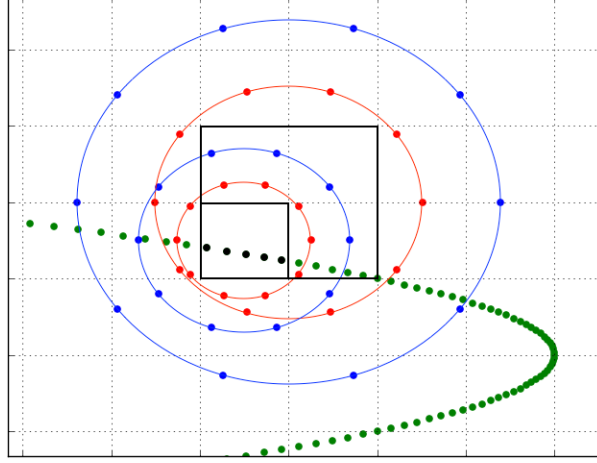


Figure 4: The upward equivalent density due to the source densities (black) in the child box computed from the upward equivalent (red) and check (blue) surfaces of the child box is translated to the upward equivalent density of the parent box, checking against its upward check (blue) and equivalent (red) surfaces.

of a box B on the same level.

$$\text{M2L: } \int_{\mathbf{y}^{B,d}} k_n(\mathbf{x}, \mathbf{y}) \phi_n^{B,d}(\mathbf{y}) r' d\mathbf{y} = \int_{\mathbf{y}^{A,u}} k_n(\mathbf{x}, \mathbf{y}) \phi_n^{A,u}(\mathbf{y}) r' d\mathbf{y} \text{ for all } \mathbf{x} \in \mathbf{x}^{B,d} \quad (20)$$

$$\text{Discretized M2L: } M_n^B \phi_n^{B,d} = M_n^A \phi_n^{A,u} \quad (21)$$

where M_n^A is the matrix of kernel evaluations of pairs of discretization points on the downward check surface of box B and the upward equivalent surface of box A , and M_n^B is the matrix of kernel evaluations of pairs of discretization points on the downward check surface of box B and the downward equivalent surface of box B .

After repeating that step, every non-leaf box has a downward equivalent density. The **L2L** operator translates the downward equivalent density of a non-leaf box A to the downward equivalent

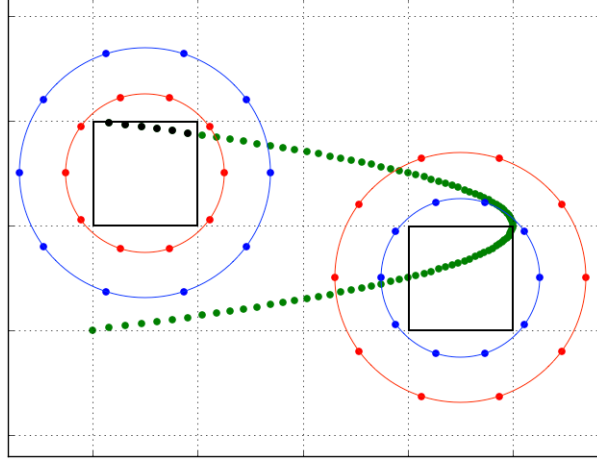


Figure 5: The upward equivalent density due to the source densities (black) in a box computed from the upward equivalent (red) and check (blue) surfaces of the child box is translated to the downward equivalent density of another box on the same level, checking against its upward check (blue) and equivalent (red) surfaces.

density of a child box B .

$$\text{L2L: } \int_{\mathbf{y}^{B,d}} k_n(\mathbf{x}, \mathbf{y}) \phi_n^{B,d}(\mathbf{y}) r' d\mathbf{y} = \int_{\mathbf{y}^{A,d}} k_n(\mathbf{x}, \mathbf{y}) \phi_n^{A,d}(\mathbf{y}) r' d\mathbf{y} \text{ for all } \mathbf{x} \in \mathbf{x}^{B,d} \quad (22)$$

$$\text{Discretized L2L: } M_n^B \phi_n^{B,d} = M_n^A \phi_n^{A,d} \quad (23)$$

where M_n^A is the matrix of kernel evaluations of pairs of discretization points on the downward check surface of box B and the downward equivalent surface of box A , and M_n^B is the matrix of kernel evaluations of pairs of discretization points on the downward check surface of box B and the downward equivalent surface of box B . After repeating this step, every box now has an upward and downward equivalent density.

Once we have these upward and downward equivalent densities for every $n = N, \dots, N$, then we can reconstruct the 3D equivalent densities ϕ_{approx} by summing as in (10). This summing can be accelerated via the FFT. Having these equivalent densities in 3D, we can complete the last step of the KIFMM algorithm by evaluating the far- and then near-field interactions.

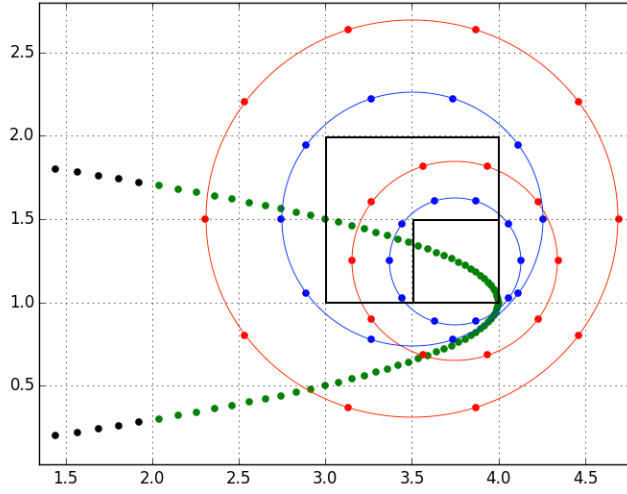


Figure 6: The downward equivalent density due to the source densities (black) in a parent box computed from the upward equivalent (red) and check (blue) surfaces of the parent box is translated to the downward equivalent density of a child box, checking against its upward check (blue) and equivalent (red) surfaces.

Note that all requirements smoothness and uniqueness of the solution to the integral equations in the KIFMM listed in §3.1.5 of [4] (mostly pertaining to the sizes and positions of the surfaces) are abided here.

3.2.2 Discretization details

The equations in §3.2.1 are discretized using p points on the equivalent and check surfaces. p is constant for all boxes, and the choice of p determines the error level of the computations. This discretization requires two steps. First, the right hand side is the evaluation of a check potential. This step checks that the potential represented by the equivalent density and the actual source densities are the same to all boxes in the far field. Then on the left hand side, we invert a Dirichlet-type boundary integral equation to obtain the equivalent density. To stably solve this equation, as in [4], we use Tikhonov regularization with regularization parameter $\alpha = 10^{-12}$. Essentially, each

translation is simply applying a series of matrices.

For example, the M2L operator is the matrix T_n^{M2L} for a mode n is obtained by solving (20):

$$\phi_n^{B,d} = \left[[\alpha I + (K_n^B)^T K_n^B] (K_n^B)^T K_n^A \right] \phi_n^{A,u} \quad (24)$$

$$\implies T_n^{M2L} = \left[[\alpha I + (K_n^B)^T K_n^B] (K_n^B)^T K_n^A \right] \quad (25)$$

3.2.3 Non-uniform translation operators

It's important to note that in the original KIFMM, translation operators only differ based on relative position and level in the hierarchical tree. This is because the kernels used there only depended the relative difference between coordinate values, e.g. $(r - r')$. As mentioned in §3.1, for the single-layer Laplace kernel, however, we are not so lucky and so we need to look for some other relationship between the translation operators.

In particular, we notice from (3) and (14) that each integral operator

$$K_n = k_n(r, z, r', z') r' = \sqrt{\frac{r'}{8\pi^3 r}} Q_{n-\frac{1}{2}} \left(\frac{r^2 + (r')^2 + (z - z')^2}{2rr'} \right) \quad (26)$$

depends on $\frac{r^2 + (r')^2 + (z - z')^2}{2rr'}$ and a constant $\sqrt{\frac{r'}{8\pi^3 r}}$. Consequently, the translation operators depend on these same values.

For example, take $\frac{r^2 + (r')^2 + (z - z')^2}{2rr'}$ and $\sqrt{\frac{r'}{8\pi^3 r}}$ as the arguments in consideration for an M2L operator from one box to a box horizontally one unit away. If we wanted to know the M2L operator from that same box to a box horizontally two units away, we could substitute $r' + 1$ for r' and see that now the operator depends on $\frac{r^2 + (r')^2 + 2r' + 1 + (z - z')^2}{2rr' + 2r}$ and $\sqrt{\frac{r'}{8\pi^3 r} + \frac{1}{8\pi^3 r}}$. In future work, we hope to find concrete relationships between the translation operators to minimize the pre-computation. Otherwise we'd have to compute every translation operator in the entire computational domain which would be expensive.

4 Implementation and Future Work

Currently, we are able to produce all equivalent densities and translation operators with programs written and tested in Python, using the single-layer 3D Laplace kernel's Fourier modes for any n . We have tested with several non-circular axisymmetric surfaces of revolution.

Much of the future work to do is in fully implementing the algorithm in Python. One key piece of this will be accelerating the matrix operations in §3 using singular value decomposition (SVD) and other acceleration techniques described in [3]. SVD is applicable in this scenario since all interactions in the far field are low rank for many kernels.

In addition, in 3.1 we mentioned that this strategy is currently only applicable to the single-layer Laplace kernel because of the analytic determination of its Fourier modes. We hope to be able to apply this strategy to other kernels as well by discretizing them as in [5].

Other implementation work that needs to be done is testing various kernels, surfaces, and potentials, to perhaps find a methodical way to determine the optimal number of discretization points p and the truncation parameter N .

Lastly, once the algorithm is completely accelerated and applicable to different kernels, performing error and computational analysis will be an important part of its evaluation as a method.

References

- [1] Cheng, H., Greengard, L., Rokhlin, V., *A Fast Adaptive Multipole Algorithm in Three Dimensions*. Journal of Computational Physics 155, (1999), 468-498.
- [2] Hao, S., Martinsson, P.G., Young, P., *An efficient and highly accurate solver for multi-body acoustic scattering problems involving rotationally symmetric scatterers*. Computers and Mathematics with Applications 69, (2015), 304-318.

- [3] Martinsson, P.G., Rokhlin, V., *An accelerated kernel-independent fast multipole method in one dimension*. SIAM Journal of Scientific Computing Vol. 29, No. 3, (2007), 1160-1178.
- [4] Ying, L., Biros, G., Zorin, D., *A kernel-independent adaptive fast multipole method algorithm in two and three dimensions*. Journal of Computational Physics 196, (2004), 591-626.
- [5] Young, P., Yao, S., Martinsson, P.G., *A high-order Nyström discretization scheme for boundary integral equations defined on rotationally symmetric surfaces*. Journal of Computational Physics 231, (2012), 4142-4159.