Fully Automating the Error-Constrained Fast Multipole Method for Multi-System, Scalar-plus-Vector Potential Laplace Problems

Ryan Anderson*, Porter Nelson, Andrew Ning

Department of Mechanical Engineering, Brigham Young University, Provo, UT, 84602, UT, USA

Abstract

The scalar-plus-vector potential N-body problem is a hurdle to be overcome in many applications. The Laplace Fast Multipole Method (FMM), originally developed by Greengard and Rohklin, reduces the computational cost of this problem from $\mathcal{O}(N^2)$ to $\mathcal{O}(N)$. However, some parameters must be tuned to navigate the tradeoff between runtime and accuracy. In this work, we present a fully self-tuning FMM. This requires fast, accurate error estimates, which are hard to come by. We derive a less conservative error bound than is typically found in the literature, and use it to dynamically choose the expansion order of each interaction subject to an error tolerance. We make some minor modifications to self-tuning dual tree traversal as found in the literature which allow us to predict the optimal leaf size. The cost of scalar-plus-vector Laplace potential problems is reduced by leveraging the Lamb-Helmholtz decomposition to reduce the number of expansions needed from 4 to 2. We generalize the self-tuning process to combine multiple source systems in a single FMM call, and include formulae for obtaining the multipole coefficients of constant vector lines, surfaces, and volumes in $\mathcal{O}(1)$ based on equivalent scalar sources. The effectiveness of the tuning strategy and the efficiency of the numerical methods are demonstrated on 2 canonical problems, as well as a vortex particle simulation of an electric VTOL aircraft. Our in-house FMM code is available open-source.

Keywords: Fast multipole method, Automated tuning, Lamb-Helmholtz decomposition, Error constrained, Regularized bodies, Quadrature to expansion

1. Introduction

The scalar-plus-vector Laplace potential problem arises in many contexts, particularly in electromagnetism [1], linear elasticity [34], and fluid dynamics [2]. In practice, it reduces to the N-body problem, because the Laplace Green's function and/or its derivatives must be either: convolved over sources in order to solve for discrete boundary element strengths; and/or evaluated at target locations to estimate force, velocity, or other field quantities. Greengard and Rohklin's Fast Multipole Method (FMM) [3] reduces the computational cost of the N-body problem from $\mathcal{O}(N^2)$ to $\mathcal{O}(N)$ and is well-suited for CPU-[10] and GPU-[26, 27, 28] parallelization, earning its status as one of ten "algorithms of the century" [5].

Bodies of finite volume (e.g. filaments, panels, volumes, or regularized particles) presents additional challenges, such as obtaining multipole coefficients. Greengard et al. divided panels into a collection of points for generating multipole expansions. In this approach, the threshold between farfield and nearfield must be chosen such that the point sources still resemble panels within an error tolerance. Additional effort was required to correct for panels that "poke" out of their cells [33]. Salloum and Lakkis approximated regularized vortex particles as point vortices when obtaining their multipole coefficients, and used the tree depth to limit the error [39]. Gumerov developed recursive $\mathcal{O}(1)$ formulae for integrating multipole coefficients of constant source filaments, panels, and volumes, as well as constant dipole panels, avoiding the discretization error of replacing panels with points at negligible computational cost [14]. The challenge remains to account for bodies that don't fit in their cells; but this can be remedied by adjusted each cell's radius to account for

*Corresponding author:
Email address: rymanderson@gmail.com (Ryan Anderson)

the finite radius of its bodies [9, 10]. It is worth mentioning that this approach requires the interaction list of each cell to be chosen dynamically, which most FMM codes do not.

The original FMM uses a rigid interaction stencil to determine which cells receive expansions [3]. Dehnen developed an adapative $\mathcal{O}(N)$ tree walk, called dual tree traversal, which evaluates expansion based on a multipole acceptance criterion (MAC) based on cell radius and separation distance rather than a rigid stencil [11, 13]. Yokota reported reduced computational cost by adjusting both the MAC and the expansion order beyond what is possible by adjusting only the expansion order [12]. The downside is that the multipole translation operators cannot be precomputed and stored if the interaction list is not known a priori. In this work, we dynamically build the interaction list, use Gumerov's analytic multipole coefficients for bodies of finite volume, and precompute portions of the translation operators to improve efficiency.

Other progress has been made to reduce the cost of FMM. For example, Gumerov et al. showed how to represent the 3-dimensional Laplace vector potential using 2 expansions (instead of 3) via the Lamb-Helmholtz decomposition, with analytic expressions of their spatial derivatives, along with the multipole coefficients for point vortices [6]. In this work, we combine the scalar potential expansion with the first component of the Lamb-Helmholtz decomposition of the vector potential, requiring only 2 expansions for both scalar and vector potential fields and their spatial derivatives. We also derive a formula for the multipole coefficients of non-point constant vortex elements. This allows us to calculate the influence of an arbitrary collection of different element types (e.g. points, filaments, or panels of monopole, dipole, or vortex sources) in a single FMM call using 2 expansions.

There is a tradeoff between computational cost and accuracy in FMM which must be tuned. In the ideal case, we select tuning parameters that minimize computational cost subject to an error tolerance. A conservative error upper bound based on the MAC was provided by Greengard, which he used to tune the expansion order [32]. One reason theoretical upper bounds may perform poorly in practice is that there may or may not be particles at the worst-case location. Dachsel developed a two-stage approach to account for this, first assuming a uniform particle distribution for tuning, and then probing the particle locations of each cell to find the recipient of the largest error and adjusting the expansion order accordingly [18]. Pringle developed a less conservative upper bound which he used to tune the expansion order of each multipole-to-local transformation, but leaving the MAC constant; he found computational savings compared to a constant expansion order [19]. Dehnen developed an even less conservative upper bound for gravitational (i.e. source) bodies to dynamically tune the MAC, but kept the expansion order constant [30]. Yokota and Barba sought to tune both the MAC and expansion order simultaneously; he reported the asymptotic behavior of both the error and the computational cost in terms of MAC and expansion order, but found that the coefficients depend too heavily on hardware and implementation for them to be useful in tuning. Instead, he tuned them empirically [26].

The leaf size (tree depth) is another tuning parameter. If it is too large (shallow), too many operations may be left to be performed directly at the leaf level; if it is too small (deep), multipole-to-local transformations dominate the cost. Dachsel used benchmarks of the multipole-to-local transformation and direct interactions to determine the optimal tree depth. Combined with the error-control mentioned in the previous paragraph, he automated FMM tuning subject to an error constraint [25], though changing the MAC (interaction list) was not considered. Yokota and Barba benchmarked the relative cost of multipole-to-local transformation, direct evaluation, and multipole evaluation, allowing them to choose the cheaper of the three, effectively ignoring all cells past the optimal depth. Multipole evaluation provided a slight cost improvement, and their strategy was effective on CPU, GPU, and hybrid architectures [35]. Salloum and Lakkis used a priori benchmarks to automatically select the optimal tree depth subject to an error tolerance, which included both expansion error and regularization error [39].

In this work, we automate the tuning process for all tuning parameters: expansion order, MAC, and leaf size, subject to a user-defined error constraint, generalized for multi-system problems. First, we describe how several different systems, including bodies of nonzero radius, can act as sources in a single FMM call. Then, we derive a less conservative error prediction that is useful for arbitrary Laplace kernels (like filaments and panels) with or without the Lamb-Helmholtz decomposition. Unlike most upper bound error predictions, it does not assume that source and target cells are the same size, making it ideal for bodies of nonzero radius or very sparse systems where cells can be shrunk. We leverage this error prediction to dynamically tune the expansion order and thereby satisfy the error tolerance. With expansion order and leaf size automatically tuned and satisfying the error constraint, tuning the MAC reduces to a 1-dimensional problem, which we

automate. We demonstrate on two canonical problems, as well as aerodynamic simulation of an electric vertical takeoff and landing passenger aircraft.

2. Algorithm Overview of the Fast Multipole Method

The FMM algorithm achieves $\mathcal{O}(n)$ scaling by leveraging four ideas: 1) a single multipole expansion expresses the influence of a cluster of source bodies; 2) many multipole expansions (converging in the farfield) can be combined into a single local expansion (converging in the nearfield); 3) multipole and local expansions can be formed very efficiently using hierarchical clustering; and 4) interactions which are too close to be evaluated using expansions must be calculated directly. We explain ideas 1-3 in the following sections in a general way (i.e. not specific to the Laplace potential). Then, we introduce the basis functions used for the Laplace potential in the rest of the paper.

2.1. Multipole Expansion

A multipole expansion expresses the influence at $\vec{x}$ of a cluster of N_b bodies over basis functions about the multipole expansion center $\vec{x}_M$. This compression is illustrated in Fig. 1, and is expressed mathematically as

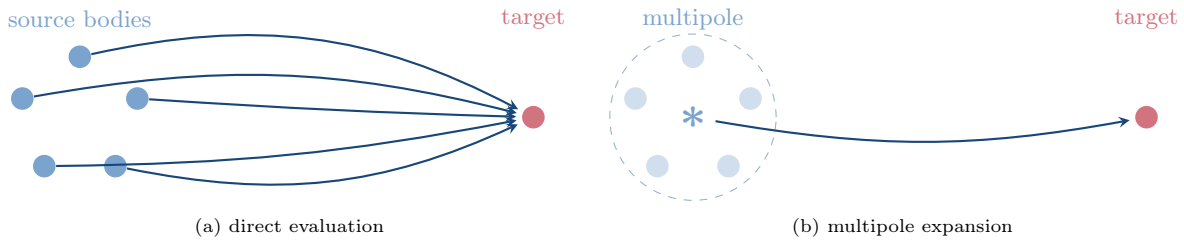
$$\phi(\vec{x}) = \sum_{j=0}^{\infty} M_j S_j(\vec{x} - \vec{x}_M) \tag{1}$$

where M_j are the multipole coefficients, S_j are the basis functions of the expansion, and the expression converges when $\vec{x}$ is farther away from $\vec{x}_M$ than the farthest body in the cluster. These coefficients are obtained by expanding each body's influence over the same basis and summing:

$$\phi(\vec{x}) = \sum_{i=1}^{N_b} \sum_{j=0}^{\infty} M_j^{(i)}(\vec{x}_i - \vec{x}_M) S_j(\vec{x} - \vec{x}_M) \tag{2}$$

$$= \sum_{j=0}^{\infty} S_j(\vec{x} - \vec{x}_M) \sum_{i=1}^{N_b} M_j^{(i)}(\vec{x}_i - \vec{x}_M) \tag{3}$$

where $M_j^{(i)}$ are the coefficients of body i and $M_j = \sum_{i=1}^{N_b} M_j^{(i)}(\vec{x}_i - \vec{x}_M)$ are the coefficients of the entire cluster. Note that the reordering of sums in Eq. 3 is only possible because the basis functions S_j are not a function of the body locations $\vec{x}_i$, and the multipole coefficients M_j are not a function of $\vec{x}$.



(a) direct evaluation (b) multipole expansion

Figure 1: A multipole expansion compresses the influence of a cluster of source bodies into a single expansion.

Truncating the infinite series after $p+1$ terms, a multipole expansion allows us to compute the influence of N_b source bodies over N_t targets in $\mathcal{O}(\max(N_b, N_t))$ rather than $\mathcal{O}(N_b N_t)$. If used for the entire N-body problem whenever targets are far enough away for multipole expansions to converge to an acceptable error tolerance, it is possible to achieve $\mathcal{O}(N \log N)$ scaling. To achieve $\mathcal{O}(N)$ scaling, we require local expansions.

2.2. Local Expansion

Local expansions differ from multipole expansions in that they converge when the evaluation point $\vec{x}$ is closer to the local expansion center $\vec{x}_L$ than the closest source body. As such, they typically employ a different basis $\{R_n(\vec{x}-\vec{x}_L)\}$:

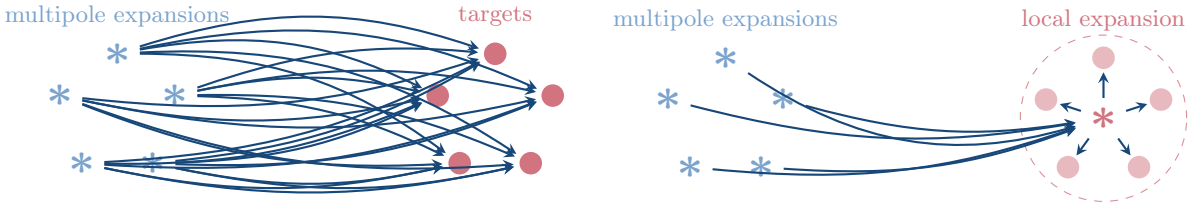
$$\phi(\vec{x}) = \sum_{j=0}^{\infty} L_j R_j(\vec{x}-\vec{x}_L) \tag{4}$$

where L_j are the local expansion coefficients. In practice, coefficients are obtained by transforming the multipole basis $\{S_j(\vec{x}-\vec{x}_M)\}$ into the local basis $\{R_j(\vec{x}-\vec{x}_L)\}$ using the multipole-to-local (M2L) transformation $T_{M2L} : \{S_j(\vec{x}-\vec{x}_M)\} \to \{R_j(\vec{x}-\vec{x}_L)\}$. When many multipole expansions are translated to the same local expansion and summed, we obtain a single local expansion representing the influence of all bodies composing each multipole expansions. The benefit of local expansions is depicted visually in Fig. 2. They are expressed mathematically as

$$\phi(\vec{x}) = \sum_{k=1}^{N_M} \sum_{j=0}^{\infty} T_{M2L}(M_j^{(k)}) R_j(\vec{x}-\vec{x}_L) \tag{5}$$

$$= \sum_{j=0}^{\infty} L_j R_j(\vec{x}-\vec{x}_L) \tag{6}$$

where $L_j = \sum_{k=1}^{N_M} T_{M2L}(M_j^{(k)})$ represents the influence of all N_M multipole expansions. In a naive implementation, all interactions are performed directly. In tree codes, multipole expansions are evaluated directly at each target, and local expansions are not used. In pure FMM implementations, multipole expansions are never explicitly evaluated at target locations; rather, they are transformed into local expansions first. In hybrid FMM-tree codes, each cell interaction uses whichever is cheaper.



(a) multipole expansions only

(b) multipole and local expansions

Figure 2: A local expansion is formed by re-expanding several multipole expansions about the center of a target cluster, thereby reducing the number of operations.

2.3. Efficient Expansion Formation and Hierarchical Clustering

There is a tradeoff between using many small body clusters vs. fewer large clusters. For multipole expansions, larger clusters are more efficient as they represent more source bodies; however, smaller clusters can be evaluated at more locations due to their smaller convergence radius. For local expansions, larger clusters are more efficient in terms of the number of targets they represent; however, smaller clusters will be able to combine the influence of more multipole expansions. For both multipole and local expansions, smaller clusters mean fewer interactions need to be evaluated directly. To navigate this tradeoff, FMM employs hierarchical clustering, where each cluster consists of a number of smaller child clusters, recursively down the hierarchy until some stopping criterion is reached. The smallest clusters are called leaves. Having several levels of cluster sizes to choose from allows us to use both large and small clusters. In this work, we use adaptive octree clustering.

FMM requires 3 steps: the upward pass, horizontal pass, and downward pass. In the upward pass, multipole expansions are formed at each leaf using Eq. 3. Rather than forming multipole expansions at each level from the source bodies explicitly, existing expansions are translated and summed at their parent cluster using the multipole-to-multipole transformation $T_{M2M} : \{S_n(\vec{x} - \vec{x}_{\text{child}})\} \to \{S_n(\vec{x} - \vec{x}_{\text{parent}})\}$. In the horizontal pass, they are transformed to local expansions using T_{M2L} at clusters far enough away to meet a separation criterion, sometimes called the multipole acceptance criterion (MAC). Note that each parent cluster encodes the influence of all its child clusters; therefore, once a transformation is performed, there is not need to transform that cluster's children to the same target (otherwise, interactions would be double-counted). Finally, beginning at the largest cluster level, local expansions are translated and summed at all child clusters using the local-to-local transformation $T_{L2L} : \{R_n(\vec{x} - \vec{x}_{\text{parent}})\} \to \{R_n(\vec{x} - \vec{x}_{\text{child}})\}$. At this point, all well-separated interactions are accounted for in the leaf level local expansions, which are evaluated at their target locations. All other interactions are computed directly.

During the horizontal pass, the set of target clusters that receive a local expansion from a given multipole expansion form the interaction list. Different FMM implementations use different interaction lists. In this work, we use an adaptation of Dehnen's $\mathcal{O}(N)$ tree walk [11] because it adapts well to sparse tree structures and arbitrary cluster configurations. As will be discussed further in Section 3.2, we resize and recenter clusters, making this an essential feature.

2.4. Laplace Potential

The solid harmonics form a natural basis for the $1/r$ kernel, and were used in the original Laplace FMM in 3 dimensions [4]. The FMM implementation of the present work employs the solid harmonics as the expansion basis, using the normalization of Epton and Dembart [15]. Then, multipole (Eq. 1) and local expansions (Eq. 4) of order p take the form

$$\phi(\vec{x}) = \sum_{n=0}^{p-1} \sum_{m=-n}^{n} M_n^m S_n^m(\vec{x}), \qquad S_n^m(\vec{x}) = \frac{(n-|m|)!}{i^{|m|} r^{n+1}} P_n^{|m|}(\cos\theta) e^{im\phi} \tag{7}$$

$$\phi(\vec{x}) = \sum_{n=0}^{p-1} \sum_{m=-n}^{n} L_n^m R_n^m(\vec{x}), \qquad R_n^m(\vec{x}) = (-1)^n \frac{i^{|m|} r^n}{(n+|m|)!} P_n^{|m|}(\cos\theta) e^{im\phi} \tag{8}$$

where r, θ, and ϕ are the radial, polar, and azimuthal coordinates of $\vec{x}$, and P_n^m are the associated Legendre polynomials. See their paper for the translation operators. In this work, we use Gumerov's efficient approach for obtaining the transformation coefficients of the point and shoot algorithm [16]. We also use Gumerov's approach for computing derivatives of the potential when evaluating expansions [6]. Due to the complexity of the algorithm, efficient FMM codes can be hard to come by, and are often taylored to specific problems. Our efficient, multi-purpose Julia code called FastMultipole[1] is open-source.

3. Methods

This section is organized as follows. In Section 3.1, we set the stage for our self-tuning algorithm by showing how the FMM can generalize to solve multi-system problems simultaneously. In Sections 3.2-3.3, we describe the different error sources to be considered in the FMM. We derive a low-cost prediction of multipole-to-local error. Because it is based on the multipole and local coefficients themselves, it can be used for pure dipoles, unlike traditional upper bounds. We describe our approach to satisfy a user-defined error tolerance ε_{tol}, thereby tuning the expansion order. Then, in Sections 3.4-3.4.2, we describe how we generalize dual tree traversal to include multiple source systems in a single FMM call, as well as our prediction of the optimal leaf size. In Section 3.5, we show how choosing the optimal MAC reduces to a 1-dimensional problem. Finally, since we leverage Gumerov's Lamb-Helmholtz decomposition of the FMM [6], we show how to obtain the multipole coefficients of constant vortex filaments, panels, or volumes of arbitrary geometry in Section 3.6.

[1] https://github.com/byuflowlab/FastMultipole.jl

3.1. Generalizing to Multi-System Problems

It is common in engineering applications for several distinct systems of bodies to interact. In electromagnetism, filaments representing wires and volumetric conductive bodies interact. In linear elasticity, the scalar potential represents bulk deformation, and the vector potential represents shear deformation. Or, in fluid dynamics, vortex volumes, panels, filaments and particles represent vorticity, source elements represent bodies of finite thickness, and dipole elements represent the effect of lift. In each case, the system induces a vector field $\vec{v}$ in terms of a scalar-plus-vector potential, or

$$\vec{v} = \nabla\phi + \nabla \times \vec{\psi} \tag{9}$$

where $\vec{v}$ could be the fluid velocity, electric field, etc. Often, code is developed to calculate the influence of each set of bodies separately, in which case interactions must be accounted for between each set of elements. This evolves into a higher-level n-body problem, where we must compute the influence of each system on every other system. However, we compute all interactions in a single FMM call by superimposing the multipole coefficients due to each system, thereby accounting for all systems at once.

In problems where the divergence and vector Laplacian vanish, $\nabla \cdot \vec{v} = 0$ and $\nabla^2\vec{v} = \mathbf{0}$, we can reduce the dimensionality of $\vec{\psi}$ from three dimensions to two under the Lamb-Helmholtz decomposition [21]:

$$\nabla \times \vec{\psi} = \nabla\varphi + \nabla \times (\vec{r}\chi) \tag{10}$$

Gumerov et al. demonstrated how to modify the FMM translation operators to compute φ and χ rather than $\vec{\psi}$, resulting in computational savings [6]. We combine the Lamb Helmholtz φ potential with our original scalar potential ϕ, effectively reducing the four dimensional potential of general problems to two:

$$\vec{v} = \nabla\phi + \nabla\varphi + \nabla \times (\vec{r}\chi) \tag{11}$$

$$= \nabla\tilde{\phi} + \nabla \times (\vec{r}\chi) \tag{12}$$

where $\tilde{\phi} = \phi + \varphi$. Now, $\tilde{\phi}$ represents contributions from both scalar and vector potentials, and χ is required to fully represent the vector potential.

Calculating the influence of multiple systems in a single FMM call is as simple as superimposing the expansion coefficients, and then computing direct interactions as usual. It is simple to do, both conceptually and in terms of implementation. However, we have found no treatment of it in the literature. We emphasize that this techniques can be applied to other FMM implementations, with or without the Lamb-Helmholtz decomposition, for scalar and or vector potentials, and regardless of the kernel used.

3.2. Error Prediction

FMM error has 4 known contributors: 1) regularization error, e.g. approximating a Gaussian regularized kernel as a singular point source in the farfield; 2) multipole expansion truncation error due to the finite expansion order; 3) local expansion truncation error due to the finite expansion order; and 4) error due to the translation of a truncated multipole expansion to a local expansion. We share how to deal with 1-3 in the following sections and neglect 4, which has been shown to be less than 30% of the expansion error for expansion orders less than or equal to 10, but is harder to predict [20]. Our goal is an order-of-magnitude accurate error prediction that allows us to automatically tune the expansion order of each multipole-to-local transformation to satisfy a user-defined absolute error tolerance ε_{tol}.

An argument can be made that a relative error tolerance would be preferred. This can be imposed in two ways–either by predicting error relative to the final target potential, or predicting error relative to the current contribution as it is calculated. The latter is easier to implement, but could lead to overly conservative expansion orders. For example, if a multipole expansion would exert a negligible influence at a target, then a relative error tolerance might require a very high expansion order for nothing. The former case is desirable, but requires an *a priori* estimate of the final potential. This can be obtained by a low-accuracy FMM call, or by remembering the potential at each body from the previous FMM call if they are known to change slowly. With that in hand, the present method can be used easily to impose a relative error tolerance.

3.2.1. Regularization Error

Regularization error occurs when bodies' influence is not represented by the $1/r$ potential. For example, a regularization factor might be applied close to singular elements (like point vortices) to eliminate the singular behavior and improve numerics. If multipole expansions are generated by assuming a true point vortex, or if the regularized function is not harmonic (and hence cannot be represented in the multipole basis), then significant error will be experienced inside the regularization region. The same phenomenon occurs for volumetric or planar sources, whose potential is harmonic only when evaluated outside their volume. For example, the gravitational potential induced by a spherical planet of constant density is harmonic outside its radius, but is linear in r inside its radius.

To remedy this, we assign an expansion radius ξ_i to the i th body equivalent to the distance at which its potential matches the $1/r$ potential within the desired error tolerance ε_{tol}. This can be expressed as finding ξ such that

$$\left|\tilde{\phi}(r) - \phi(r)\right| \leq \varepsilon_{tol} : r \geq \xi \tag{13}$$

where $\tilde{\phi}(r)$ is the regularized influence and $\phi(r)$ is the singular influence approximated by the FMM. Then, we generalize the approach by Deng et al. [9] to shrink (or grow) and recenter each cell until it fully encapsulates all member sources. If bodies have a finite radius (like panels), we set ξ_i equal to the geometric radius. If bodies have a regularized kernel, we calculate ξ based on $\left|\tilde{\phi}(r) - \phi(r)\right|$ directly rather than the size of the panel.

3.2.2. Truncation Error

The scalar potential is expanded in terms of a multipole expansion as

$$\phi(\vec{r}) = \sum_{n=0}^{\infty} \sum_{m=-n}^{n} \phi_n^m X_n^m(\vec{r}) \tag{14}$$

where X_n^m are the irregular solid harmonics S_n^m for a multipole expansion and the regular solid harmonics R_n^m for a local expansion. The error of a p-truncated expansion is then the sum of the truncated terms:

$$\varepsilon_\phi = \sum_{n=p}^{\infty} \sum_{m=-n}^{n} \phi_n^m X_n^m(\vec{r}) \tag{15}$$

$$\approx \sum_{m=-p}^{p} \phi_p^m X_p^m(\vec{r}) \tag{16}$$

Due to the oscillatory nature of the spherical harmonics, this error is highly dependent on the evaluation point, and can vary significantly if the evaluation point is perturbed by even a small amount[2]. To remove this sensitivity in favor of an upper bound for a multipole expansion, Dehnen substituted ϕ_n^m with an expression of the multipole power, leveraging the fact that the norm of spherical harmonic coefficients of the same degree is invariant under rotation of the coordinate system [30]. We use the same technique as follows.

Consider $X_n^m = S_n^m$ (multipole expansions). Separating the spherical harmonics from the solid harmonics, we have

$$\phi_n^m S_n^m(\vec{r}) = \frac{\phi_n^m \tilde{S}_n^m}{|\vec{r}|^{n+1}} Y_n^m(\vec{r}) \tag{17}$$

[2]Often in the literature, an upper bound has been obtained by replacing the Legendre polynomial factor of X_n^m with its upper bound of unity. However, this becomes unnecessarily conservative [30].

where $\tilde{S}_n^m$ is a normalizing factor that converts the normalization of S_n^m to the orthonormal normalization of Y_n^m. For our normalization,

$$\tilde{S}_n^m = \sqrt{\frac{4\pi(n+|m|)!(n-|m|)!}{(2n+1)}} \tag{18}$$

Noting that $\sum_{m=-n}^{n} \phi_n^m \tilde{S}_n^m$ has a norm that is invariant under rotation, we define $\mathcal{M}_n$ to be rotationally invariant as

$$\mathcal{M}_n \equiv \sqrt{\sum_{m=-n}^{n} \left[\phi_n^m \tilde{S}_n^m\right]^2} \tag{19}$$

At this point, we depart from Dehnen's method. Consider rotating the coordinate system such that the z axis aligns with the point of highest multipole error located a distance r away. Then, the $|m| > 0$ terms vanish, and

$$\varepsilon_\phi \lesssim \frac{\mathcal{M}_p}{\tilde{S}_p^0} \frac{(p)!}{r^{p+1}} \tag{20}$$

Because $\mathcal{M}_n$ is rotationally invariant, we only need to know the radial distance r of the max-error location– not the angular location, which is difficult to predict. To be conservative, we choose r as the distance from the multipole expansion center to the closest point in the target cell.

The preceding theory is sufficient for predicting the multipole expansion error, but not the local expansion error. It is the authors' understanding that Dehnen assumed the two to be equal, which has been shown to be a good assumption when multipole and local cells are the same size [19]. This is not guaranteed in our implementation, however, as will be discussed in Section 3.4. So additional treatment is needed to account for local expansion error, as follows.

Consider a vector ϕ_n^m of coefficients of R_n^m with

$$(L_\phi)_n^m R_n^m(\vec{r}) = \phi_n^m \tilde{R}_n^m Y_n^m(\vec{r}) r^n \tag{21}$$

Reconstructing the normalization like we did before, we have:

$$i^{|m|}(-1)^m \sqrt{\frac{(2n+1)(n-|m|)!}{4\pi(n+|m|)!}} \tilde{R}_n^m = (-1)^n \frac{i^{|m|}}{(n+|m|)!} \tag{22}$$

$$\tilde{R}_n^m = \sqrt{\frac{4\pi}{(2n+1)(n+|m|)!(n-|m|)!}} \tag{23}$$

which implies the rotation-invariant norm:

$$\mathcal{L}_n = \sqrt{\sum_{m=-n}^{n} \left[\phi_n^m \tilde{R}_n^m\right]^2} \tag{24}$$

Then, rotating into a z-aligned coordinate system with the point of maximum error such that $|m| > 0$ terms vanish, we arrive at the following for a local expansion:

$$\varepsilon_\phi \lesssim \frac{\mathcal{L}_p}{\tilde{R}_p^0} \frac{r^p}{(p)!} \tag{25}$$

We can conservatively choose r as the distance from the local expansion center to the farthest corner of the cell.

In practice, we can precompute and store $\tilde{R}_n^m$ and $\tilde{S}_n^m$ once for the entire simulation. We note that $\mathcal{M}_n$ and $\mathcal{L}_n$ each require $\mathcal{O}(n)$ operations to calculate. The largest cost of this error estimate is computing the multipole and local coefficients for $n = p$.

It is often more helpful to control the error in the induced vector field $\vec{v}$ from Eq. 9 rather than the potential itself. Upper bound estimates can be derived as

$$|\vec{\varepsilon}_R| \lesssim \sqrt{3}\frac{\mathcal{L}_p^{(\tilde{\phi})}}{\tilde{R}_p^0}\frac{r^{p-1}}{(p-1)!} + \sqrt{3}\frac{\mathcal{L}_p^{(\chi)}}{\tilde{R}_p^0}\frac{r^p}{(p)!} \tag{26}$$

$$|\vec{\varepsilon}_S| \lesssim \sqrt{3}\frac{\mathcal{M}_{p-1}}{\tilde{S}_{p-1}^0}\frac{(p)!}{r^{p+1}} + \sqrt{3}\frac{\mathcal{M}_p^{(\chi)}}{\tilde{S}_p^0}\frac{p!}{r^{p+1}} \tag{27}$$

where $|\vec{\varepsilon}_R|$ is the magnitude of the vector error experienced by a local expansion, and $|\vec{\varepsilon}_S|$ is for a multipole expansion. The derivation can be found in Appendix A.

3.3. Dynamic Expansion Order

To efficiently constrain the error of the FMM, we seek the smallest expansion order such that the error tolerance is satisfied according to the error prediction of the previous section. The upward and downward passes do not introduce significant error [19], so we focus on the horizontal pass, where multipole expansions are transformed into local expansions and accumulated at target cells. We adopt the "point-and-shoot" approach for efficient transformation of multipole and local coefficients developed by White and Head-Gordon. As depicted in Fig. 3, we rotate the coordinate system of a multipole expansion such that the z axis points to the desired local expansion center. This is because z axis translation is more efficient than an arbitrary translation vector, requiring $\mathcal{O}(p^3)$ operations for the multipole-to-local transformation instead of $\mathcal{O}(p^4)$. After performing the z-axis transformation, we back-rotate the coordinate system to its original orientation. [38]

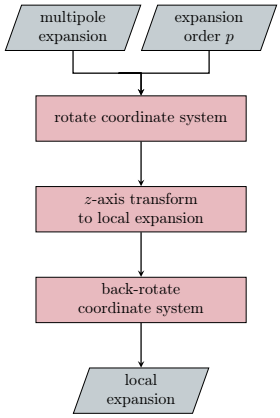


Figure 3: Point-shoot-rotate algorithm for multipole-to-local transformation.

First, a max expansion order $p_{\max}$ is selected for the upward pass. This will represent the largest possible expansion order we can perform later. In the horizontal pass, we calculate the coefficients $\tilde{\phi}_n^m$ and χ_n^m of a single degree n at a time, and stop when the error tolerance is reached or we reach $n = p_{\max}$, as shown in Fig. 4. This is straightforward for rotating the multipole coefficients, as each rotation requires only the coefficients of the same degree n. For the z-axis transformation, however, all coefficients of the same order m are required. We still calculate them one n at a time, but note that after the error tolerance is satisfied, each coefficient ϕ_n^m will need to be updated by coefficients of the same m of degree greater than n. With that in mind, we perform a z-axis transformation, and then check the error predicton. If it is not satisfied, we increment n and iterate until we reach $n = p_{\max}$. If it is satisfied, we choose the expansion order $p \leftarrow n - 1$,

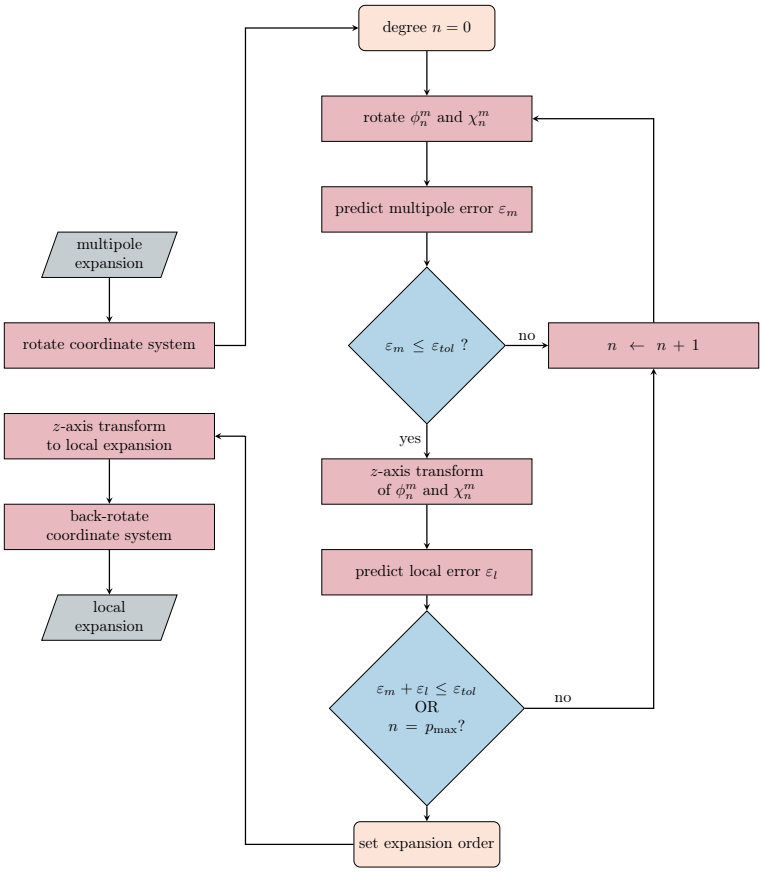


Figure 4: Error-constrained dynamic expansion order algorithm.

and proceed to correct coefficients with $n < p$. Since the z axis transformation is a small portion of the cost, we simply repeat it once the p is known. Finally, we back-rotate the coordinate system.

We point out that the method calculates multipole coefficients of order n to predict the error behavior for coefficients of order $n-1$. In practice, it is poosible to use all calculated coefficients by using all order n coefficients to increase our accuracy for a negligible increase in cost. The only extra cost is using the rotation matrices (which we have already calculated) to back-rotate the local coefficients of order n.

3.4. Auto-tuning the Leaf Size with Dual Tree Traversal

In dual tree traversal, a multipole acceptance criterion (MAC) determines a distance beyond which expansions are allowed. There are several definitions of the MAC in the literature, but we use the one developed by Warren and Salmon [10], as used by Yokota and Barba [12] and Dehnen [30]. That is, given a source cell with radius ρ_S and a target cell with radius ρ_T, separated by a distance d, we accept the expansion if

$$\text{MAC} > \frac{\rho_S + \rho_T}{d} \tag{28}$$

Beginning at the root of the octree, expansions may be used between cells that satisfy the MAC. If two cells do not satisfy the MAC, one of them is subdivided into octants and the method recurses until the number of bodies inside a branch is less than a pre-determined minimum leaf size. Unlike the traditionally rigid interaction list, this approach does not require consistent symmetry in the size and relative location of clusters. This is ideal not only for non-uniform body distributions, but also for bodies of nonzero radius, in which case we resize and recenter the clusters. The auto-tuning mechanism functions as follows. If the MAC is satisfied, the cheapest of the following occurs:

1. the expansion is transformed to a local expansion at the target cell
2. interactions are computed directly between source and target bodies
3. the expansion is evaluated at the target bodies within the target cell
4. a local expansion is formed from the source bodies at the target cell

Our implementation is depicted in Fig. 5 where C_D is the cost of direct calculation between two particular cells, C_{M2L} is the cost of the multipole-to-local transformation, and N_{target} and N_{source} are the numbers of bodies in the target and source cell, respectively. Our approach has a few minor differences. First, Yokota and Barba showed that bullet 3 is rarely preferred, and Dehnen reported that 4 is rarely preferred, so we omit these from our implementation. However, our approach is easily amenable to including these if desired. The main difference is the use of the threshold leaf size $\hat{s}^{(i)}$ to determine which operation is the cheapest, as will be discussed subsequently. In short, this allows the optimal leaf size to be known a priori. Another difference is the choice of which cell to subdivide. Traditionally, this defaults to the cell with the larger radius; in our implementation, we choose the cell that contains more bodies, regardless of its physical size. This choice is likely to produce a more balanced tree structure. We also point out that our error prediction is uniquely qualified for this approach because it maintains accuracy for different source/target cell sizes, unlike other methods known to the authors [19, 30].

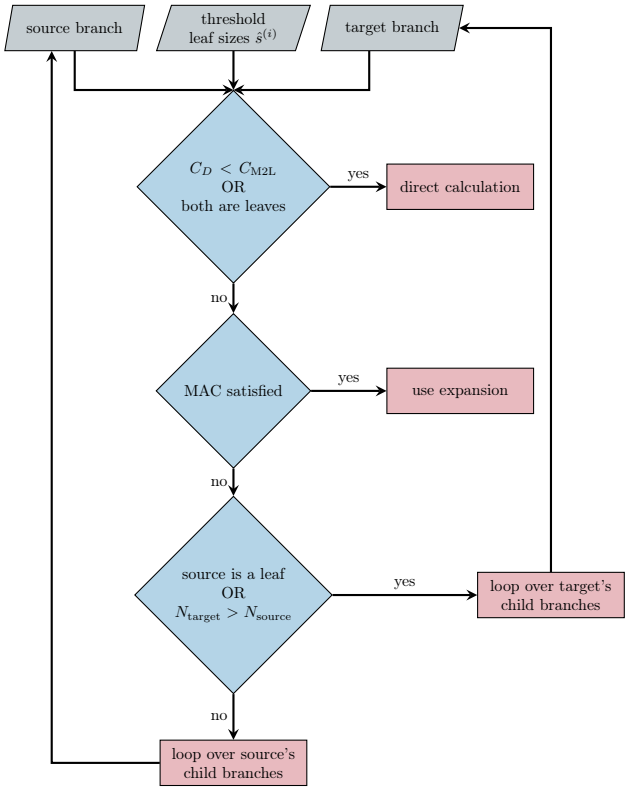


Figure 5: Our implementation of self-tuning dual tree traversal. C_D is the cost of direct calculation between two particular cells, C_{M2L} is the cost of the multipole-to-local transformation, and N_{target} and N_{source} are the numbers of bodies in the target and source cell, respectively.

If the expansion order is known, estimating the cost of the expansion operators is straightforward. In our implementation, however, we do not know the expansion order a priori (see Section 3.3). So, we run the FMM to obtain average benchmarks. This could lead to sub-optimal performance if the chosen expansion orders have a high variance, but it performs well in practice, as demonstrated in Section 4. An alternative approach would be to constrain the error based on the MAC instead, keeping the expansion order fixed as demonstrated by Dehnen [30].

3.4.1. Optimal Leaf Size for a priori Tree Creation

The first step in an FMM call is to create the tree. Then, during tree traversal, we push the indices of branches to receive direct or multipole-to-local transformations to two vectors, respectively. After completing the tree traversal, direct and multipole-to-local transformations are performed using for loops rather than recursion, which is typically more efficient. We predict the optimal leaf size s as follows.

We define a threshold leaf size $\hat{s}$ as the number of bodies at which the cost of a multipole expansion equals that of a direct evaluation, or

$$\hat{s} = \sqrt{\frac{C_{exp}}{C_D}} \tag{29}$$

where C_D is the cost of a single direct interaction, and C_{exp} is the average cost of an expansion. We estimate the former by benchmarking during an FMM call. We could estimate the latter by benchmarking the horizontal pass and dividing by the number of multipole-to-local transformations; however, this ignores the overhead cost required to form the octrees, the upward pass, and the downward pass. We find more optimal tuning parameters when we sum benchmarks for all of these and then divide by the number of multipole-to-local transformations. The downside is that the benchmarks are dependent on the tuning parameters, making this an iterative process in the worst case. This is a non-issue, however, when the FMM is used for a time evolution [8], or during an iterative solution process [36], in which the distribution of bodies changes only slightly from timestep to timestep. In either case, we can borrow benchmarks with parameters that are close to the optimum during the previous timestep or iteration for free. In one-off applications, or for the first timestep, we often find good convergence in just three FMM calls: one to determine the max expansion order needed for the upward and downward passes, one to determine the leaf size, and one to ensure the expansion order still satisfies ε_{tol} after changing the leaf size.

When considering whether or not to use an expansion, we consider the number of bodies in the source cell N_S and the number of bodies in the target cell N_T. If the number of bodies is small, it will be cheaper to evaluate interactions directly. If the number of bodies is large, it will be cheaper to use the expansion. The break-even point occurs when

$$\frac{N_S N_T}{\hat{s}^2} = 1 \tag{30}$$

If the preceding expression is less than unity, direct interactions are preferred. Otherwise, the expansion is used (see Fig. 5).

The ideal leaf size is just small enough that the algorithm never runs out of branches. Because we always subdivide the branch containing more bodies, it is reasonable to assume that $N_T \geq N_S$. Relaxing this assumption by a small amount, we choose the source and target tree leaf sizes $s_S = s_T = \hat{s}/2$.

3.4.2. Optimal Leaf Size for Multiple Source Systems

We desire to generalize the approach of the previous section for multiple source systems in a simultaneous FMM solve (see Section 3.1). Because all source systems contribute to the same expansions, a single source tree is constructed containing all source systems. We assign variables a superscript index $^{(i)}$ referring to the ith source system, and define $\hat{s}^{(i)}$ for each source system using the system of equalities

$$C_{exp} = \left(\hat{s}^{(i)}\right)^2 C_D^{(i)} \tag{31}$$

where $C_D^{(i)}$ is the cost of a direct evaluation of the ith source system on a single target. As before, $C_D^{(i)}$ and C_{exp} can be obtained by benchmarking an FMM call.

We generalize the break-even point of Eq. 30 as

$$C_{exp} = N_T \sum_i N_S^{(i)} C_D^{(i)} \tag{32}$$

$$= N_T \sum_i N_S^{(i)} \frac{C_{exp}}{(\hat{s}^{(i)})^2} \tag{33}$$

$$N_T \sum_i \frac{N_S^{(i)}}{(\hat{s}^{(i)})^2} = 1 \tag{34}$$

If the expression is less than unity, direct interactions are preferred; otherwise, expansions are used.

When forming the tree, we base the tree depth on the cost break-even point of Eq. 34. It is reasonable to assume that N_T is never smaller than $\sum_i N_S^{(i)}$ because we choose to subdivide whichever cell contains more bodies in the tree walk, as discussed in Section 3.4. Then, it is conservative to stop subdividing the source tree when[3]

$$\min_i(\hat{s}^{(i)}) \sum_i \frac{N_S^{(i)}}{(\hat{s}^{(i)})^2} \leq 1 \tag{35}$$

By a similar logic, we can stop subdividing the target tree when

$$N_S \leq \min_i \hat{s}^{(i)} \tag{36}$$

Note that if only one source system is used, the stopping criterion of Eq. 35 reduces to the traditional leaf size definition:

$$N_S \leq \hat{s} \tag{37}$$

3.5. Choosing the Optimal MAC

Now that we have automated the choice of expansion order, satisfied an error tolerance, and automated the choice of optimal leaf size, the only parameter remainding is the MAC. Dehnen suggests that a MAC of 0.4 is the theoretical optimum [30], but we have found that this is not always true in practice (see Section 4.1.5). Fortunately, this is a simple 1-dimensional problem which we solve easily by testing a range of reasonable candidate values and choosing the one that results in the lowest computational cost. The suggested procedure in Section 3.4.1 costs roughly 3 FMM calls per candidate MAC. We suggest performing this once for a typical timestep or iteration and reusing the same MAC throughout the simulation.

3.6. Multipole Coefficients for the Lamb-Helmholtz Decomposition

In order to use FMM under the Lamb-Helmholtz decomposition, we require multipole coefficients in terms of ϕ and χ. Gumerov derived the multipole coefficients for point vortices [6]. We show how to extend his derivation to constant vortex filaments, panels, and volumes.

Gumerov showed that the multipole coefficients $(M_\phi)_n^m$ and $(M_\chi)_n^m$ for a point vortex of strength $\vec{\omega}$ at location $\vec{x}$ are

$$\phi_n^m(\vec{r}_s) = \frac{1}{n+1}\left[(\omega_x - i\omega_y)\frac{n-m+1}{2}R_n^{m-1}(-\vec{x}) - (\omega_x + i\omega_y)\frac{n+m+1}{2}R_n^{m+1}(-\vec{x}) - i\omega_z m R_n^m(-\vec{x})\right] \tag{38}$$

$$\chi_n^m = -\frac{1}{n}\left[\frac{1}{2}(\omega_y + i\omega_x)R_{n-1}^{m-1}(-\vec{x}) - \frac{1}{2}(\omega_y - i\omega_x)R_{n-1}^{m+1}(-\vec{x}) - \omega_z R_{n-1}^m(-\vec{x})\right],\ \chi_0^0 = 0 \tag{39}$$

[3] In practice, we relax the assumption that $N_T \geq \sum_i N_S^{(i)}$ by changing the RHS from 1 to 0.5.

We obtain the multipole coefficients by integrating over the element:

$$\vec{u}(\vec{r}_t) = \int \nabla \times \frac{\vec{\omega}}{|\vec{r}_t - \vec{r}_s|} dS \tag{40}$$

$$= \sum_{n=0}^{\infty} \sum_{m=-n}^{n} \tilde{g}_n^m \ \nabla \times [\vec{\omega} S_n^m(\vec{r}_t)] \tag{41}$$

where $\tilde{g}_n^m = \int R_n^m(-\vec{x}) dS$ are the multipole coefficients of a constant source point, filament, panel, or volume sharing the same geometry (e.g., scalar potential only). Then, using the same process as Gumerov [6] to obtain Eqs. 38-39, we arrive at:

$$\phi_n^m(\vec{r}_s) = \frac{1}{n+1} \left[(\omega_x - i\omega_y) \frac{n-m+1}{2} \tilde{g}_n^{m-1} - (\omega_x + i\omega_y) \frac{n+m+1}{2} \tilde{g}_n^{m+1} - i\omega_z m \tilde{g}_n^m \right] \tag{42}$$

$$\chi_n^m(\vec{r}_s) = -\frac{1}{n} \left[\frac{1}{2}(\omega_y + i\omega_x) \tilde{g}_{n-1}^{m-1} - \frac{1}{2}(\omega_y - i\omega_x) \tilde{g}_{n-1}^{m+1} - \omega_z \tilde{g}_{n-1}^m \right], \ \chi_0^0 = 0 \tag{43}$$

which are the multipole coefficients for a constant vortex filament, panel, or volume. To calculate $\tilde{g}_n^m$, we use the recursive $\mathcal{O}(1)$ procedure published by Gumerov [14].

4. Results

4.1. Point Source and Point Vortex Example

To test the methods, we first consider two canonical problems. For the first problem, we randomly place point sources according to a uniform distribution in the unit cube with strengths sampled from a uniform distribution between 0 and 1. Then, we normalize the strengths such that the mean gradient $\nabla\phi$ is unity. The induced potential of a point source of strength m located at $\vec{r}$ can be expressed as

$$\phi(\vec{r}_t) = \frac{m}{|\vec{r}_t - \vec{r}|} \tag{44}$$

For the second problem, we randomly place point vortices in the unit cube with x, y, and z components of their vortex strength randomly sampled from a uniform distribution between -1 and 1. Then, we normalize the vortex strengths such that the mean curl magnitude $|\nabla \times \psi|$ is unity. The influence can be expressed as

$$\psi(\vec{r}_t) = \frac{\vec{\omega}(\vec{r}_t - \vec{r})}{4\pi |\vec{r}_t - \vec{r}|^2} \tag{45}$$

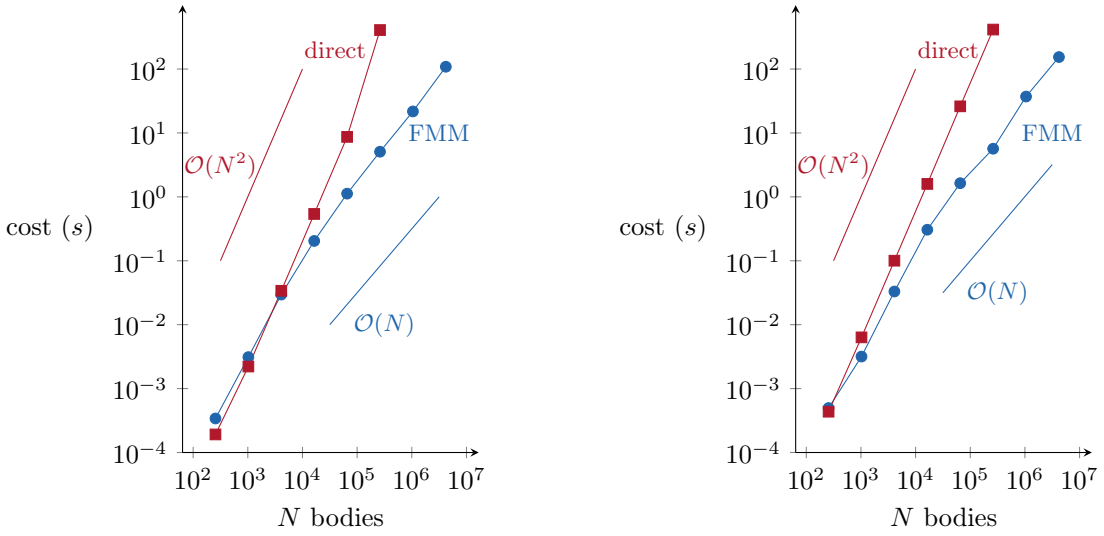
$$\nabla \times \vec{\psi}(\vec{r}_t) = \frac{\vec{\omega} \times (\vec{r}_t - \vec{r})}{4\pi |\vec{r}_t - \vec{r}|^3} \tag{46}$$

Note that because neither point sources nor point vortices are regularized nor do they occupy finite volume, so we set the expansion radius (described in Section 3.2.1) $\xi = 0$ and consider only the truncation error.

4.1.1. Verification of $\mathcal{O}(N)$ Scaling

To verify that our FMM implementation works properly, we benchmark the FMM on a single thread at a series of system sizes for the point source and point vortex systems, respectively. The expansion order is fixed to $p = 4$, and the leaf size is fixed at $s = 50$. We then compute interactions directly for comparison and plot the results in Fig. 6. Based on the slope of the log-log plots, we verify that for large enough problems, the FMM cost scales as $\mathcal{O}(N)$ and the direct interactions scale as $\mathcal{O}(N^2)$. We also note that for very small problems, the FMM scaling more closely resembles $\mathcal{O}(N^2)$. This is likely because most leaf level branches are too close for expansions at this leaf size, meaning direct interactions dominate the cost. The cross-over point where the direct cost exceeds the FMM cost occurs at ~ 2000 point sources, or ~ 250 point vortices.

We also note that the cost of the direct vortex interactions is roughly 3 times that of the direct source interactions. This makes sense because the vortices induce a 3-dimensional potential, whereas the sources induce a 1-dimensional potential. The difference in cost when running the FMM, however, is only 10-15% higher. We attribute this to the cost of calculating the rotation matrices, which is the dominant cost, and can be reused for both dimensions of the Lamb-Helmholtz potential.



(a) point source system

(b) point vortex system

Figure 6: $\mathcal{O}(N)$ scaling of the FMM demonstrated.

4.1.2. Combined Scalar-plus-Vector Potential Verification

To verify that our method of combining scalar and vector potentials under two expansions is convergent (see Section 3.1), we calculate the induced vector field with FMM and compare it to that obtained directly (e.g. without FMM). We define the vector error ε_{max} here and throughout Section 4 as

$$\varepsilon_{\max} = \arg\max_i |\vec{v}_i - \hat{v}_i|_2 \tag{47}$$

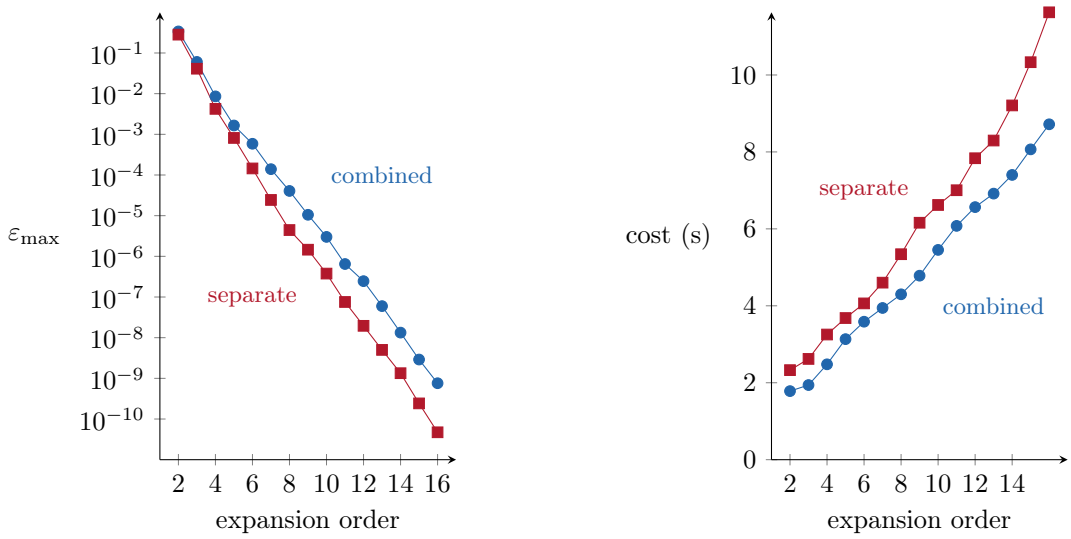
where $\vec{v}_i$ is predicted directly, $\hat{v}_i$ is predicted with the FMM, and the subscript i denotes the i th body.

In Fig. 7a, we show that the combined scalar-plus-vector error converges with increased expansion order. Note that we fix the leaf size at 50 and the MAC at 0.5. The error seems to converge faster when expansions are performed separately, but this is likely a side-effect of keeping the leaf size constant when combining the FMM calls. A deeper tree is created for the combined system when the leaf size is kept constant. Then, a higher percentage of interactions are handled by expansions, introducing more truncation error. Since the bodies are more densely arranged in the combined system, the same leaf size results in smaller leaf-level cells. This means that expansions are used for a larger fraction of the computation, leading to larger error.

The cost of computing these interactions separately is compared to the cost of performing them together in Fig. 7b. We observe a roughly 25% decrease in cost due to combining the expansions. It has been theorized that for a well-tuned FMM, the cost of expansions is roughly half the overall cost, with the other half due to direct interactions. For roughly equal-sized systems, the cost of computing interactions separately is 2 FMM calls of roughly equal cost. Since the same direct interactions must be performed in either case, but we can combine many of the multipole-to-local transformations, we would predict a cost of 1.5 FMM calls, which matches the 25% savings we observe.

4.1.3. Expansion Error

To verify the error prediction derived in Section 3.2, we consider the source system and vortex system separately. During the horizontal pass of an FMM call, we use the error estimate from Section 3.2 to predict

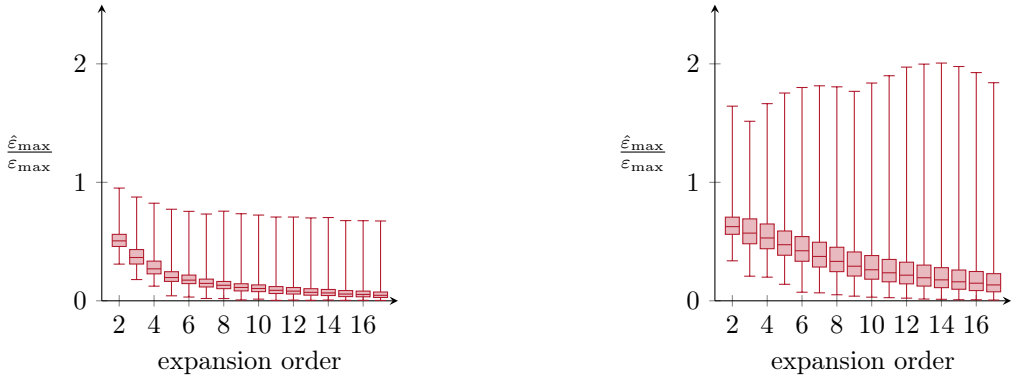


(a) error convergence

(b) cost reduction

Figure 7: Simultaneous computation of the scalar-plus-vector potential of superimposed point sources and point vortices. Leaf size and MAC are held constant at 50 and 0.5, respectively.

$\varepsilon_{\max}$ for each multipole-to-local transformation. First, we evaluate the local expansion at all member bodies in the target cell. Then, we calculate the influence at the same locations using direct interactions due to the member bodies of the source cell, and calculate the error. We store the largest error ε_{max} for each multipole-to-local transformation. In Fig. 8, we plot the ratio of $\varepsilon_{\max}$ to the predicted upper-bound $\hat{\varepsilon}_{\max}$ over various expansion orders for the source system and vortex system, respectively. An upper tail of 1 indicates a perfect upper bound prediction, which we observe consistently for the source system. The vortex system appears to under-predict by a factor of 2 or so. We attribute the long lower tails to fluctuations in the error as a function of the polar angle, caused by oscillations in the Legendre polynomials composing our basis functions. The median and lower tails tend to drop as the expansion order is increased, indicating that the probability of realizing the error upper-bound decreases as the expansion order increases. This makes sense, since the Legendre polynomial frequency increases with the expansion order, making it less likely to probe near a maximum value. Note that we fix the leaf size at 50 and the MAC at 0.5.



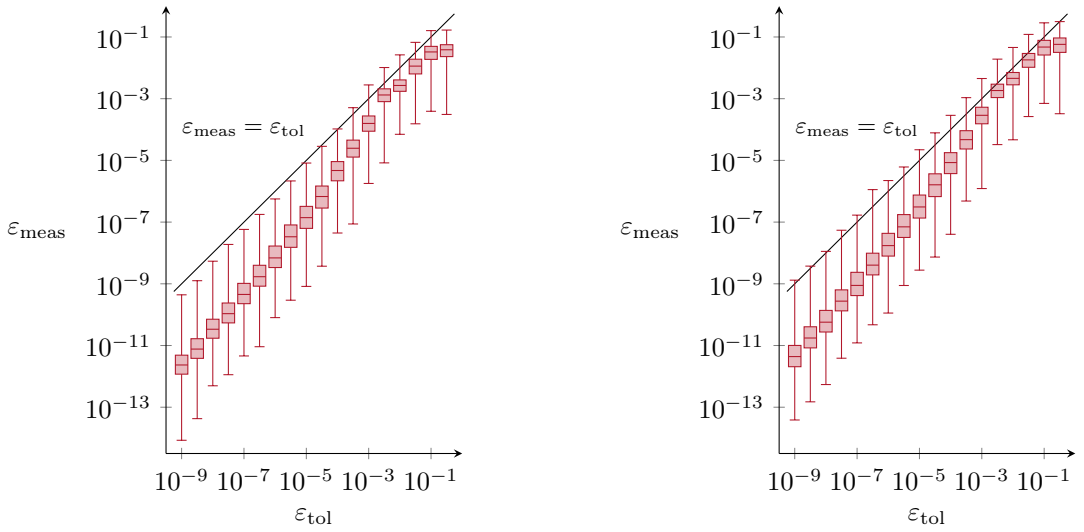
(a) point sources

(b) point vortices

Figure 8: Accuracy of the error prediction method of Section 3.2 at all multipole-to-local interactions of a single FMM call at various expansion orders. Note that an upper tail at 1 indicates a perfect upper bound prediction. Leaf size and MAC are fixed at 50 and 0.5, respectively.

4.1.4. Dynamic Expansion Order

To test the dynamic expansion order selection, we run the FMM over $\varepsilon_{tol} \in [10^{-9}, 10^{-1}]$ with a leaf size of 50 and MAC of 0.5. We also compute the interactions directly (without FMM) and use them to calculate the maximum absolute velocity error $\varepsilon_{\max}$. We plot the distribution of $\varepsilon_{\max}$ for each tolerance for the source and vortex systems in Figs. 9a and 9b, respectively. The dotted line is placed at $y = x$ to represent the error ceiling; a perfect error prediction would result in upper tails at, but not exceeding, this line. In the point source system, the $\varepsilon_{\max}$ is conservatively constrained for $\varepsilon_{\text{tol}} < 10^{-4}$. Above that, the error rises to its worst case of a factor of 3 at $\varepsilon_{\text{tol}} = 10^{-2.5}$. In the vortex system, $\varepsilon_{\max}$ is constrained within a factor of between 1 and 3 for $\varepsilon_{\text{tol}} < 10^{-4}$. In that sense, the algorithm is less conservative for the vortex system, consistent with the behavior seen in Figs. 8a and 8b. Like the source system, the vortex system also experiences its worst constraint violation near $\varepsilon_{\text{tol}} = 10^{-2.5}$. We hypothesize this to be related to additive error from several multipole expansions transforming to the same local expansion, which we don't account for in our method. Since the spherical harmonics of higher degree experience higher frequency oscillations, it makes sense that the error of incoming expansions would tend to cancel at higher expansion orders; conversely at lower expansion orders, they would be more likely to aggregate.



(a) point sources

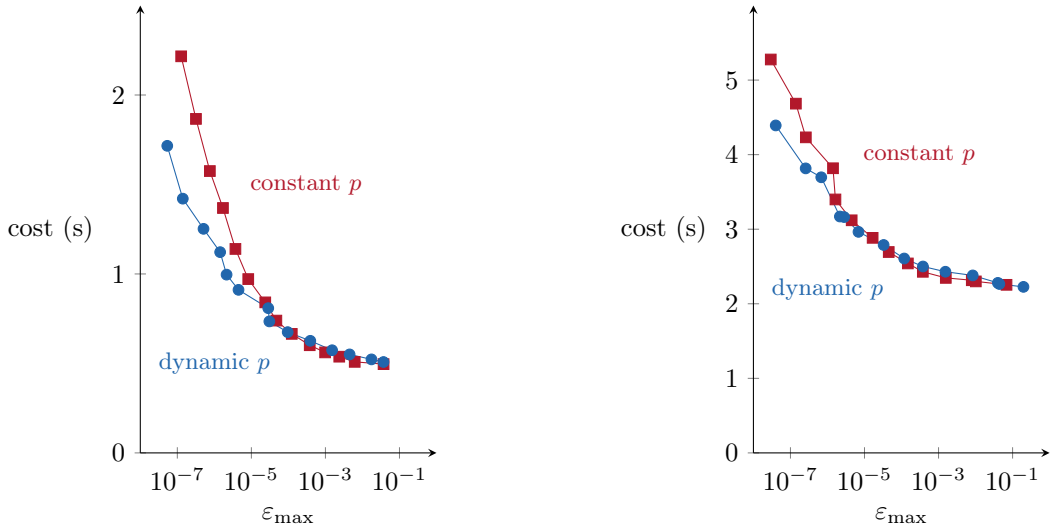
(b) point vortices

Figure 9: Distribution of $\varepsilon_{\max}$ after using the dynamic expansion order from Section 3.3 over a range of ε_{tol}. An upper tails or 1 indicates the error tolerance was imposed perfectly. The leaf size and MAC are set to 50 and 0.5, respectively.

Ideally, constraining the error should not result in additional computational cost. To measure the efficiency of the algorithm, we run the FMM on the same situation as in Fig. 9, but fixing the expansion order. Then, measure $\varepsilon_{\max}$ for each expansion order, and run the FMM again using the dynamic expansion order to constrain the error. For maximum efficiency, we use the higher expansion order $p \leftarrow n$ rather than $p \leftarrow n-1$ discussed in Section 3.3. We compare the computational cost vs. ε_{max} in Fig. 10. It is worth noting that the dynamic expansion order is never more than 5% slower than a fixed expansion order, and is always faster for tolerances tighter than 10^{-5}. The corresponding reduction in cost is more significant as ε_{tol} decreases. This makes sense because the cost of the multipole-to-local transformation scales as $\mathcal{O}(p^3)$, translating to greater savings when higher expansion orders are used.

4.1.5. Optimal Leaf Size and MAC

To verify the optimal leaf size and MAC prediction of Sections 3.4-3.4.2 and 3.5, we first perform a brute-force parameter space exploration to get a sense for what the optimal parameters should be. Then, we compare our auto-tuned parameters to a manual parameter optimization. For the manual optimization, we start at the auto-tuned parameters, and perturb the leaf size and MAC to see if any neighboring parameters result in a lower cost. If they do, we accept that point and repeat until we find a local minimum.

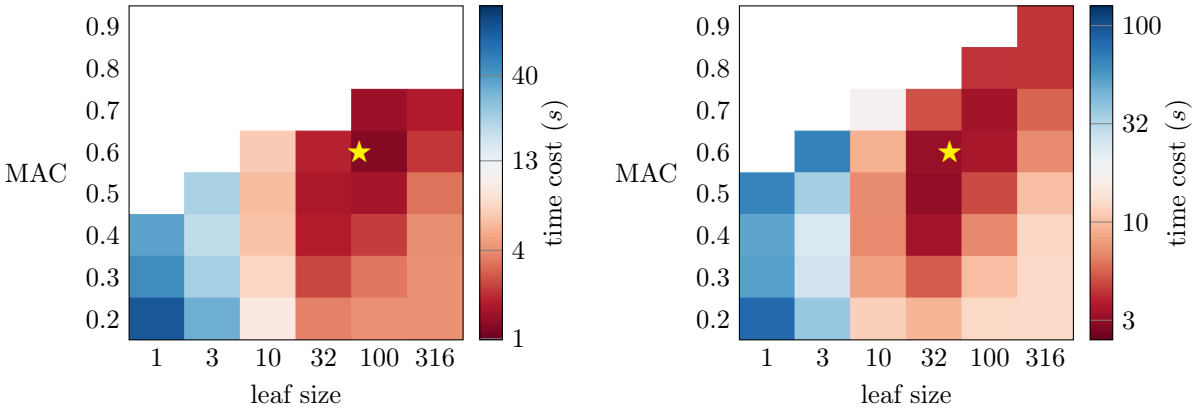


(a) point sources

(b) point vortices

Figure 10: Reduction in cost due to a dynamically selected expansion order compared to keeping it constant. The leaf size and MAC are set to 50 and 0.5, respectively.

In Figs. 11a-11b, we show the time cost for various leaf size and MAC parameters for the point source and point vortex systems, respectively. The dynamic expansion order was used with $\varepsilon_{\text{tol}} = 10^{-6}$ and $p_{\max}$ chosen as the largest requested expansion order, and white space indicates that ε_{tol} could not be satisfied for $p \leq 36$. The yellow star indicates the auto-tuned parameters. We note how in some cases, a very small change in one of the parameters could result in significantly increased computational cost. It is also worth noting that the optimal parameters are very close to violating the error constraint, highlighting the importance of a good error prediction.



(a) point sources

(b) point vortices

Figure 11: Effect of leaf size and MAC on FMM cost with $\varepsilon_{tol} = 10^{-6}$. The yellow star indicates the auto-tuned parameters, and white space indicates ε_{tol} could not be satisfied for $p_{\max} \leq 36$.

In Tables 1 and 2, we compare the auto-tuned parameters with the result of a manual grid search for the point source and vortex point systems, respectively. When interpreting this data, it is worth considering the discrete relationship of both leaf size and MAC on the performance of uniformly distributed bodies. Because we divide each cell by a factor of 8 at a time, there could be a range of leaf sizes that result in the same tree, and therefore the same computational cost. Similarly, there are a range of MAC values that lead to the same interaction list. With that in mind, expansion order and resulting computational cost are better

comparison metrics. Expansion orders match within 1, and computational costs match within 4% and 7% for the point source and point vortex problems, respectively.

Table 1: Auto-tuned parameters with $\varepsilon_{\text{tol}} = 10^{-6}$ for the point source system compared to optimal parameters obtained by manual grid search.

	leaf size	MAC	expansion order	cost (s)
auto-tuned	66	0.6	15	1.21
optimal	60	0.575	14	1.17

Table 2: Auto-tuned parameters with $\varepsilon_{\text{tol}} = 10^{-6}$ for the point vortex system compared to optimal parameters obtained by manual grid search.

	leaf size	MAC	expansion order	cost (s)
auto-tuned	44	0.6	18	2.50
optimal	55	0.6	17	2.35

4.2. Passenger eVTOL Aircraft Example

As a real-world engineering example, we model the flight of the Joby S4 passenger eVTOL aircraft. We take geometry from the OpenVSP Airshow[4] and generate a CFD surface mesh using OpenVSP[5]. We model wings and fuselage using 42,618 uniform source-plus-normal doublet panels in FLOWPanel[6]. Note that the potential induced by a uniform source panel can be expressed by

$$\phi_{\text{source}}(\vec{r}_t) = \frac{\sigma}{4\pi} \int \frac{dS}{|\vec{r}_t - \vec{r}|} \tag{48}$$

where σ is the source strength, and $\vec{r}_t$ is the evaluation point. Likewise for a uniform normal double panel,

$$\phi_{\text{doublet}}(\vec{r}_t) = \frac{\mu}{4\pi} \hat{n} \cdot \nabla \int \frac{dS}{|\vec{r}_t - \vec{r}|} \tag{49}$$

where μ is the doublet strength and $\hat{n}$ is the unit normal vector to the surface.

Since panels have a nonzero radius, care must be taken to ensure that panels don't "poke" outside their cells. We treat the panel centroid as its position in the FMM, and assign it a radius equal to distance from the centroid to the farthest vertex. Then, we shrink/grow and recenter cells such that panels are completely contained before computing the interaction list, as discussed in Section 3.2.1.

4.2.1. Regularized Vortex Filament Rotors and Kutta Condition

Rotors are modeled using the actuator line model in FLOWUnsteady [22], and are converted into 702 vortex filaments in this example. We desire to regularize the induced velocity near the filament while exactly matching the singular kernel elsewhere to make the FMM error convergent. So, we regularize the induced velocity of vortex filaments as

$$\vec{v}(\vec{r}_t) = \frac{\Gamma\ (\vec{r}_t - \vec{r}_1) \times (\vec{r}_t - \vec{r}_2)}{f_\sigma\left(|\vec{x}_r - \vec{x}_1||\vec{x}_r - \vec{x}_2| + (\vec{x}_r - \vec{x}_1) \cdot (\vec{x}_r - \vec{x}_2)\right)} \left(\frac{1}{f_\sigma\left(|\vec{x}_r - \vec{x}_1|\right)} + \frac{1}{f_\sigma\left(|\vec{x}_r - \vec{x}_2|\right)} \right) \tag{50}$$

[4]https://airshow.openvsp.org/vsp/LtyRrSrYXRhQlDkYwPDX
[5]https://openvsp.org
[6]https://github.com/byuflowlab/FLOWPanel.jl

where Γ is the vortex strength, $\vec{x}_1$ and $\vec{x}_2$ are the endpoints of the vortex, and the regularization function f_σ is defined as

$$f_\sigma(r) = \begin{cases} r + (r-\sigma)^2, \ r < \sigma \\ r \text{ otherwise} \end{cases} \tag{51}$$

where σ is the filament core radius. Vortex filaments are also used to enforce the Kutta condition at the trailing edge of wings, with strengths Γ equal to the difference between trailing edge dipole panel strengths such that the trailing edge vorticity vanishes.

Since filaments have both a finite radius and a regularized kernel, we set their radius ξ as the sum of the half-length and the core radius σ. Note that the regularized velocity kernel of Eq. 50 matches the singular kernel exactly when probing at a distance greater than σ from the closest point on the filament. This will guarantee that no regularization error will exist if multipole expansions are used farther than σ from the filament.

4.2.2. Regularized Vortex Particle Wake

Wakes are modeled within FLOWUnsteady using vortex particles with the Gaussian regularization selected, meaning their induced velocity is Eq. 46 multiplied by a regularization function $q_\sigma(\vec{r}_t)$, with the regularization function equal to

$$q_\sigma(\vec{r}_t) = \text{erf}\left(\frac{|\vec{r}_t - \vec{r}|}{\sigma\sqrt{2}}\right) - \sqrt{\frac{2}{\pi}}\frac{|\vec{r}_t - \vec{r}|}{\sigma} e^{-|\vec{r}_t - \vec{r}|^2/(2\sigma^2)} \tag{52}$$

where σ is the particle smoothing radius. Vortex particles themselves are a Lagrangian discretization of the incompressible viscous Navier-Stokes equations using Alvarez and Ning's more stable reformulation [23] as implemented in FLOWVPM[7]. We use 105,410 vortex particles in this example.

We desire to control the regularization error of vortex particles in the FMM. The expansions used in the FMM assume no regularization (a pure $1/r$ kernel); for proper error control, this discrepancy must be accounted for in the FMM. As discussed in Section 3.2.1, we define a radius ρ for each particle that will be used to adjust the cell radius. It is defined such that the regularized velocity $\tilde{\vec{v}}$ differs from the non-regularized velocity $\vec{v}$ by less then the error tolerance ε_{tol} when evaluated at $|\vec{r}_t - \vec{r}| \geq \rho$:

$$\varepsilon_{tol} \geq \left|\vec{v} - \tilde{\vec{v}}\right| \tag{53}$$

$$\geq \left| \frac{\vec{\omega} \times (\vec{r}_t - \vec{r})}{4\pi|\vec{r}_t - \vec{r}|^3} \left(1 - \text{erf}\left(\frac{|\vec{r}_t - \vec{r}|}{\sigma\sqrt{2}}\right) + \sqrt{\frac{2}{\pi}}\frac{|\vec{r}_t - \vec{r}|}{\sigma} e^{-(|\vec{r}_t - \vec{r}|)^2/(2\sigma^2)}\right)\right| \tag{54}$$

$$= \frac{\omega\rho}{4\pi\rho^3}\left(1 - \text{erf}\left(\frac{\rho}{\sigma\sqrt{2}}\right) + \sqrt{\frac{2}{\pi}}\frac{\rho}{\sigma} e^{-\rho^2/(2\sigma^2)}\right) \tag{55}$$

$$0 = \frac{4\pi\sigma^2\varepsilon_{tol}}{\omega}\left(\frac{\rho}{\sigma}\right)^2 + \text{erf}\left(\frac{1}{\sqrt{2}}\frac{\rho}{\sigma}\right) - \sqrt{\frac{2}{\pi}}\frac{\rho}{\sigma} e^{-(\rho/\sigma)^2/2} - 1 \tag{56}$$

where we go from Eq. 54 to Eq. 55 by noting $\max_{\vec{r}_t} [\vec{\omega} \times (\vec{r}_t - \vec{r})] = \omega\rho$. Eq. 56 is a 1-dimensional root finding problem in ρ/σ with the solution bounding by 0 on the left. On the right, it is bounded by replacing the erf term with its minimum value of 0, and the exponential factor $e^{-(\rho/\sigma)^2/2}$ with 1 and solving the resulting quadratic. The resulting right bound is

$$\frac{\rho}{\sigma} = \frac{\omega}{8\pi\varepsilon_{tol}\sigma}\left(\sqrt{\frac{2}{\pi}} + \sqrt{\frac{2}{\pi\sigma^2} + 16\frac{\pi\varepsilon_{tol}}{\omega}}\right) \tag{57}$$

[7] https://github.com/byuflowlab/FLOWVPM.jl.git

which can be solved efficiently for ρ using a bracketing method like Brent's method. ρ can then be used to adjust the cell radius to constrain the regularization to be less than ε_{tol}.

4.2.3. Timestepping Procedure

Each timestep of the aircraft simulation can be divided into 3 steps. First, the influence of the wake on itself and the aircraft is evaluated (wake-on-all). Second, the strengths of panels and filaments composing the aircraft are computed based on the wake influence. If the number of panels is large, an iterative solution method where FMM approximates the matrix-vector product without ever forming the interaction matrix may be used (panels-on-panels). Third, the influence of panels and filaments acting on themselves and the wake (aircraft-on-all) is calculated before stepping forward in time. The full aircraft and wake is visualized in Fig. 12, with individual components shown in Fig. 13.

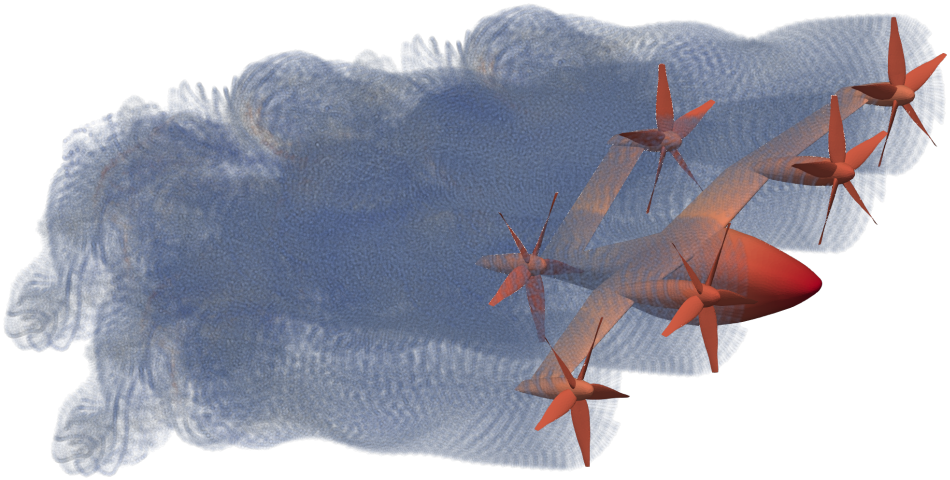


Figure 12: Visualization of the fully-assembled Joby aircraft.

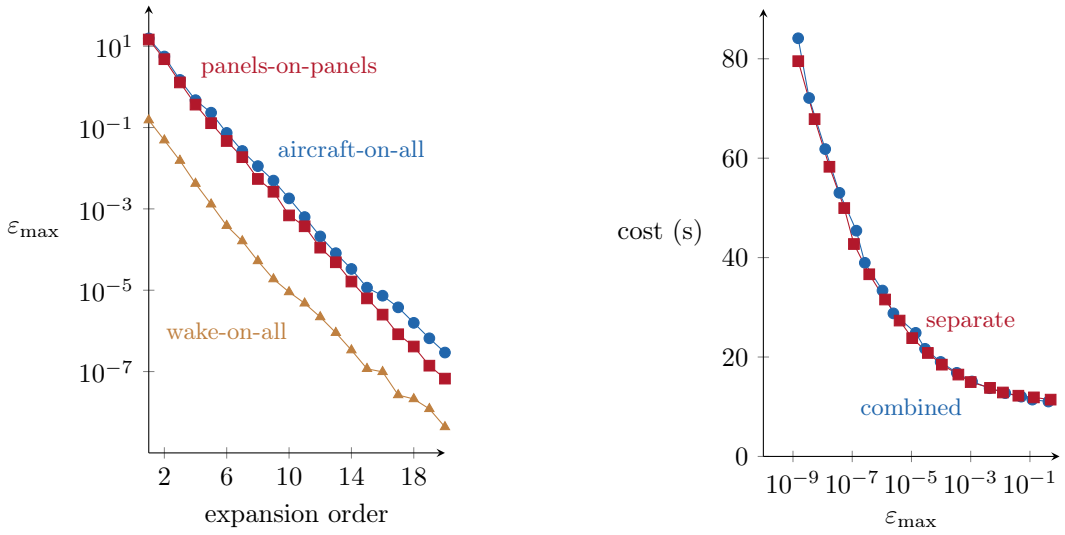
(a) vortex particle wake (b) vortex filament rotors and wing trailing edge (c) constant source panels (d) constant normal dipole panels

Figure 13: Visualization of each component of the Joby aircraft.

4.2.4. Combined Scalar-plus-Vector Potential Verification

We have already verified the combined scalar-plus-vector potential method in Section 4.1.2. We have not, however, verified our approach to control the error of regularized or nonzero-radius bodies discussed in Section 3.2. We demonstrate error convergence in Fig. 14a for each of the three interactions discussed in the previous section: wake-on-all, panels-on-panels, and aircraft-on-all. Note that the aircraft-on-all error converges slightly slower than the panels-on-panels error. This makes sense because the aircraft consists of both panels and vortex filaments.

In Fig. 14b, we compare the cost of performing the aircraft-on-all interaction one system at a time compared to combining the expansions. In this case, the costs are identical within a few percent for a given max error, indicating the cost benefit from combining expansions we saw for point sources and point vortices



(a) The maximum absolute velocity error converges for regularized vortex particles (wake-on-all), regularized bodies of non-zero radius (panels-on-panels), and simultaneous evaluation of regularized bodies of non-zero radius (aircraft-on-all).

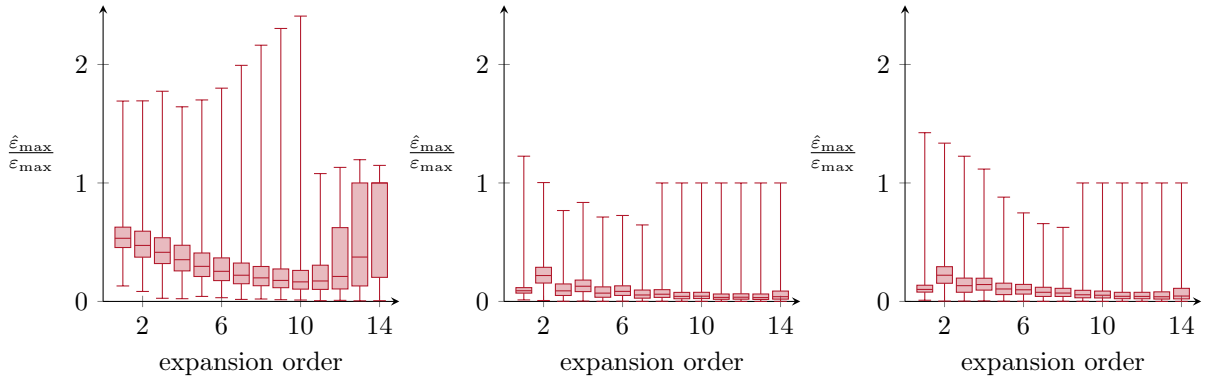
(b) Combining the expansions of aircraft-on-all into a single FMM call is only slightly faster than calculating interactions individually.

Figure 14: Cost comparison of the vehicle-on-all interaction in a single multi-system FMM call vs. two separate FMM calls.

in Fig. 7b doesn't always hold. This makes sense, considering that we use roughly 50 times more panels than filaments. Because the cost of panel interactions is already much larger than the cost of filament interactions, combining the expansions doesn't help much.

4.2.5. Error Prediction Verification

Now, we seek to verify the error prediction for panels and filaments. In Fig. 15, we report statistics for $\hat{\varepsilon}_{\max}/\varepsilon_{\max}$ for the wake-on-all, panels-on-panels, and aircraft-on-all interactions, which we generate by randomly sampling 100,000 multipole-to-local interactions. As seen in the figure, the method successfully predicts the upper bound within a factor of about 2 across the board. Note that all absolute values less than 10^{-12} are set to 10^{-12} before calculating statistics to avoid reporting discrepancies due to machine precision.



(a) wake-on-all

(b) panels-on-panels

(c) aircraft-on-all

Figure 15: Accuracy of the error prediction from Section 3.2.2 at all multipole-to-local interactions of a single FMM call at various expansion orders. Note that an upper tail at 1 indicates a perfect upper bound prediction. Leaf size is fixed at 10 for the particle field and panels, and at 35 for filaments. MAC is fixed at 0.4.

4.2.6. Dynamic Expansion Order

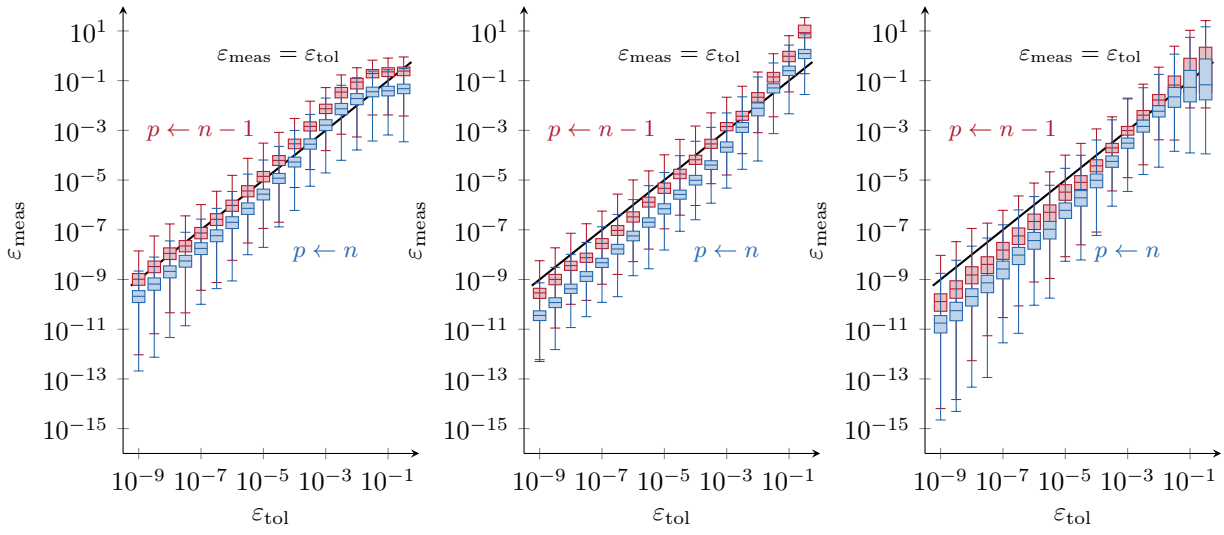
The accuracy of the dynamic expansion order selection for all three interactions is plotted in red in Fig. 16, and can be categorized into two regimes. The first regime occurs at the finest tolerances. Here, the upper tail of the observed error is within a factor between 5 and 20 of the tolerance. At a certain tolerance, we cross into the second regime where this factor begins to grow. For the wake-on-all interaction, the tolerance is exceeded by a factor of 30 at a tolerance of 10^{-5}. For panels-on-panels and aircraft-on-all, this doesn't occur until a tolerance of 10^{-2}. Interestingly, these both correspond to an expansion order of about 9 in Fig. 14a.

In the first regime, we attribute the factor of error to several source cells accumulating at a single target cell. Since the error of a local expansion tends to be largest at the 8 corners of the cell, the error due to the 8 source cells nearest the corners of the target could accumulate. Multiplying 8 by the factor of 2 observed in Fig. 15 results in a factor of 16, which matches our observations.

In the second regime, we attribute the additional factor of error to approximating the error by just one term in the infinite series of Eq. 15. For coarser tolerances, and therefore lower expansion orders, it is more likely that the $n = p + 1$ term in Eq. 15 could be larger than the $n = p$ term, which is the only one we used. In other words, it is plausible for a configuration of bodies to have a large quadrapole moment (i.e., $n = 2$) while having a small dipole moment ($n = 1$). If we relied on the dipole moment to predict the error, we would underpredict the error. However, this is less likely to happen for higher order moments, which is likely the reason for the two regimes observed. This would also explain why the coarse regime is more pronounced for the aircraft-on-all interaction, because vortex filaments have highly asymmetric moments. This might also explain why we don't see large errors for coarse tolerances of the wake-on-all interaction, since the wake consists of point vortices. The fact that the factor of error corresponds to the same expansion order (as mentioned two paragraphs ago) also supports this hypothesis.

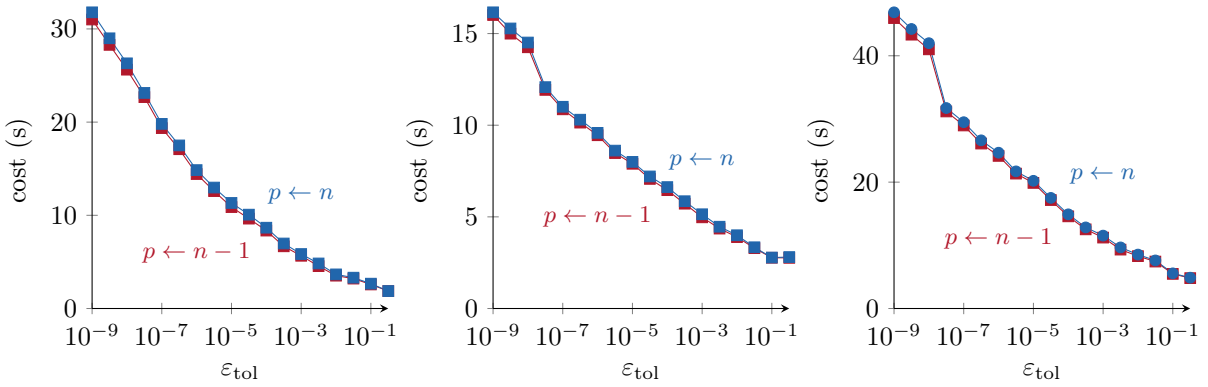
Of note, the wake-on-all maximum error appears to converge to a single value for tolerances coarser than 10^{-2}, unlike the panels-on-panels and aircraft-on-all interactions which seem to continually grow larger. This is likely because the wake-on-all interaction satisfies the error tolerance at the worst-case locations with the smallest allowable expansion order.

To improve the accuracy without sacrificing cost, we increase the expansion order by one as explained in Section 3.3, which is plotted in blue. This reduces the error factor by about an order of magnitude across the board in the first regime. The benefit is reduced in the second regime, likely for the same reason that the error increases there. The cost impact of increasing the expansion order is less than 4% at worst as shown in Fig. 17.



(a) wake-on-all (b) panels-on-panels (c) aircraft-on-all

Figure 16: Accuracy of the dynamic expansion order selection versus requested error tolerance for wake-on-all, panels-on-panels, and aircraft-on-all interactions. Leaf size and MAC were auto-tuned.



(a) wake-on-all (b) panels-on-panels (c) aircraft-on-all

Figure 17: The cost of increasing the expansion order by 1 under the dynamic expansion order is less than 4% in all cases.

4.2.7. Optimal Leaf Size and MAC

We verify the auto-tuned leaf size and MAC by comparing to a manual parameter optimization like we did in Section 4.1.5. In Tables 3-5, we compare the auto-tuned parameters to optimal parameters obtained by manual grid search, where leaf size was tested in multiples of 5. The MAC was tested in multiples of 0.05 for both the grid search and the auto-tune. For the wake-on-all interaction, the parameters match exactly. For the panels-on-panels and aircraft-on-all interactions, the auto-tuned cost is within 7% and 4%, respectively, of the grid search.

Table 3: Auto-tuned parameters for the wake-on-all interaction compared to the optimal parameters obtained by manual grid search.

	leaf size	MAC	expansion order	cost (s)
auto-tuned	20	0.5	15	14.4
optimal	20	0.5	15	14.4

Table 4: Auto-tuned parameters for the panels-on-panels interaction compared to optimal parameters obtained by manual grid search.

	leaf size	MAC	expansion order	cost, seconds
auto-tuned	11	0.4	16	9.4
optimal	20	0.45	18	8.8

Table 5: Auto-tuned parameters for the aircraft-on-all interaction compared to optimal parameters obtained by manual grid search.

	panels leaf size	filament leaf size	MAC	expansion order	cost (s)
auto-tuned	12	79	0.45	18	23.1
optimal	20	60	0.45	18	22.4

5. Conclusion

In this work, the fast multipole method (FMM) was generalized for scalar-plus-vector potential, multi-system, N-body problems in a single FMM call. The Lamb-Helmholtz decomposition was used to reduce

the number of expansions required from 4 to 2. Error due to using regularized kernels and bodies of nonzero radius was mitigated by recentering and shrinking/growing each cell based on the radius or regularization function. A less conservative error estimate was developed for the potential and the norm of its gradient, and was used to satisfy an error tolerance by choosing the expansion order for each multipole-to-local interaction dynamically. The leaf size tuning parameter was tuned automatically based on benchmarks of a single FMM call. This makes it possible to re-tune every timestep of time-evolving N-body systems at no additional cost, since benchmarks from the previous timestep can be used for tuning the current step. Since both the leaf size and the expansion order are automatically tuned under this scheme, tuning the multipole acceptance criterion simplifies to a 1-dimensional root find, which is automated. In other words, all parameters of the FMM are automatically tuned.

The methods developed were first tested on two canonical examples: randomly placed point sources or vortices in the unit cube. The error method predicted the maximum error within a factor of 2 in all cases, and the autotuned parameters resulted in a computational cost within 5% of the optimal parameters found by manual grid search. The design space of tuning parameters was plotted, showing that order-of-magnitude cost savings can be obtained by appropriate tuning. Choosing the expansion order dynamically resulted in a 15% reduction in computational over a constant expansion order for a fixed error tolerance of 10^{-7}, though this cost reduction grew smaller as the error tolerance increased until it vanished at a tolerance of roughly 10^{-5}. When both point sources and point vortices were combined into a single multi-system problem, it was shown that combining the expansions under the Lamb-Helmholtz decomposition resulted in computational savings of roughly 25% for any given expansion order and a constant leaf size.

Finally, a passenger electric vertical takeoff and landing aircraft was modeled using a combination of constant source panels, constant normal dipole panels, regularized vortex filaments, and regularized vortex particles. Three interactions were considered: wake-on-all, panels-on-panels, and aircraft-on-all, to represent practical use cases. The maximum error was successfully constrained within an order of magnitude most of the time, though a factor of 100 was observed for very coarse tolerances. By leveraging data already available within an FMM call, we were able to increase the expansion order by one for up to an order of magnitude improvement in accuracy at less than a 4% cost penalty. The auto-tuned parameters matched the manually found optimal parameters exactly for the wake-on-all interaction. For the panels-on-panels and aircraft-on-all interactions, the resulting computational cost was within 7% and 4%, respectively.

It is hoped that the methods developed here will reduce the user effort required to leverage FMM in scientific computing. The FMM code developed for this work was written in the Julia language and is available open-source on Github[8] and the Julia registry.

Appendix A. Error Upper Bound Derivation

Here, we derive Eqs. 26 and 27, which are upper bound predictions for the magnitude of the error in the vector field $\vec{v}$ induced by the scalar-plus-vector potential field under the Lamb-Helmholtz decomposition. In other words,

$$\vec{v}(\vec{r}) = \nabla\tilde{\phi} + \nabla \times (\vec{r}\chi) \tag{A.1}$$

where $\tilde{\phi}$ and χ are expanded in the FMM to expansion order p as

$$\tilde{\phi}(\vec{r}) = \sum_{n=0}^{p-1} \sum_{m=-n}^{n} \tilde{\phi}_n^m X_n^m(\vec{r}) \tag{A.2}$$

$$\chi(\vec{r}) = \sum_{n=0}^{p-1} \sum_{m=-n}^{n} \chi_n^m X_n^m(\vec{r}) \tag{A.3}$$

[8] https://github.com/byuflowlab/FastMultipole.jl

where X_n^m are the irregular solid harmonics for a multipole expansion or the regular solid harmonics for a local expansion. The multipole ($X_n^m = S_n^m$) and local ($X_n^m = R_n^m$) vector magnitude truncation error is due to terms neglected in the infinite series:

$$|\vec{\varepsilon}_X| \leq \varepsilon_{\nabla\tilde{\phi}} + \varepsilon_{\nabla\times(\vec{r}\chi)} \tag{A.4}$$

$$\varepsilon_{\nabla\tilde{\phi}} = \left| \sum_{n=p}^{\infty} \sum_{m=-n}^{n} \tilde{\phi}_n^m \nabla X_n^m(\vec{r}) \right| \tag{A.5}$$

$$\approx \left| \sum_{m=-p}^{p} \tilde{\phi}_p^m \nabla X_p^m(\vec{r}) \right| \tag{A.6}$$

$$\varepsilon_{\nabla\times(\vec{r}\chi)} = \left| \sum_{n=p}^{\infty} \sum_{m=-n}^{n} \chi_n^m \left(\nabla \times \vec{r} X_n^m(\vec{r})\right) \right| \tag{A.7}$$

$$\approx \left| \sum_{m=-p}^{p} \chi_p^m \left(\nabla \times \vec{r} X_p^m(\vec{r})\right) \right| \tag{A.8}$$

In the next two subsections, we will develop an estimate for ε under local and multipole expansions, respectively. We point out that while these results work equally with or without the Lamb-Helmholtz decomposition; we need only drop the terms pertaining to the χ potential to remove it.

Appendix A.1. Local Vector Truncation Error

It can be shown that spatial derivatives of the solid harmonics can be expressed in terms of other solid harmonics, allowing us to eliminate the ∇ operator from Eqs. A.6 and A.8. The derivation is beyond the scope of this work, but we use Gumerov's results for a local expansion [6]:

$$v_k = \sum_{n=0}^{\infty} \sum_{m=-n}^{n} v_{kn}^m R_n^m(\vec{r}) \tag{A.9}$$

$$v_{xn}^m = \frac{1}{2}\left[i\tilde{\phi}_{n+1}^{m-1} + i\tilde{\phi}_{n+1}^{m+1} + (n-m)\chi_n^{m+1} - (n+m)\chi_n^{m-1} \right] \tag{A.10}$$

$$v_{yn}^m = \frac{1}{2}\left[\tilde{\phi}_{n+1}^{m-1} - \tilde{\phi}_{n+1}^{m+1} + i(n-m)\chi_n^{m+1} + i(n+m)\chi_n^{m-1} \right] \tag{A.11}$$

$$v_{zn}^m = -\tilde{\phi}_{n+1}^m - im\chi_n^m \tag{A.12}$$

We estimate $\varepsilon_{\nabla\tilde{\phi}}$ by taking the first neglected term in the truncated expansion. Note that $n = p$ is not the first neglected term because v_{kn}^m involves coefficients of degree $n + 1$. Rather,

$$\varepsilon_{\nabla\tilde{\phi},k} \approx \sum_{m=-p+1}^{p-1} \left(v_{k,p-1}^m\right)_{\tilde{\phi}} R_{p-1}^m(\vec{r}) \tag{A.13}$$

$$\leq \frac{\mathcal{L}_p^{(\tilde{\phi})}}{\tilde{R}_p^0} \frac{r^{p-1}}{(p-1)!} \tag{A.14}$$

$$\varepsilon_{\nabla\tilde{\phi}} = \sqrt{\varepsilon_{\nabla\tilde{\phi},x}^2 + \varepsilon_{\nabla\tilde{\phi},y}^2 + \varepsilon_{\nabla\tilde{\phi},z}^2} \tag{A.15}$$

$$\lesssim \sqrt{3}\frac{\mathcal{L}_p^{(\tilde{\phi})}}{\tilde{R}_p^0} \frac{r^{p-1}}{(p-1)!} \tag{A.16}$$

where the superscript $(\tilde{\phi})$ denotes the coefficients used to compute $\mathcal{L}_p$ as per Eq. 24.

We predict the $\nabla \times (\vec{r}\chi)$ local error as

$$\varepsilon_{\hat{i}_k \cdot \nabla \times (\vec{r}\chi)} \approx \sum_{m=-p}^{p} \left(v_{k,p}^{m}\right)_{\chi} R_p^m(\vec{r}) \tag{A.17}$$

$$\leq \frac{\mathcal{L}_p^{(\chi)}}{\tilde{R}_p^0} \frac{r^p}{(p)!} \tag{A.18}$$

$$\varepsilon_{\nabla \times (\vec{r}\chi)} = \sqrt{\varepsilon^2_{\nabla \times (\vec{r}\chi),x} + \varepsilon^2_{\nabla \times (\vec{r}\chi),y} + \varepsilon^2_{\nabla \times (\vec{r}\chi),z}} \tag{A.19}$$

$$\lesssim \sqrt{3} \frac{\mathcal{L}_p^{(\chi)}}{\tilde{R}_p^0} \frac{r^p}{(p)!} \tag{A.20}$$

Substituting Eqs. A.13 and A.17 into Eq. A.4, we obtain

$$|\vec{\varepsilon}_R| \lesssim \sqrt{3} \frac{\mathcal{L}_p^{(\tilde{\phi})}}{\tilde{R}_p^0} \frac{r^{p-1}}{(p-1)!} + \sqrt{3} \frac{\mathcal{L}_p^{(\chi)}}{\tilde{R}_p^0} \frac{r^p}{(p)!} \tag{A.21}$$

which is our estimate of the upper bound of the magnitude of the vector error of a local expansion, and the second term can be ignored if the Lamb-Helmholtz decomposition is not used.

Appendix A.2. Multipole Vector Truncation Error

For the multipole $\vec{v}$ error, we desire an expression like Eq. A.9, but in the S_n^m basis. We begin with Gumerov's derivative expressions [6]:

$$D_x S_n^m = \frac{i}{2}\left(S_{n+1}^{m+1} + S_{n+1}^{m-1}\right) \tag{A.22}$$

$$D_y S_n^m = \frac{1}{2}\left(S_{n+1}^{m+1} - S_{n+1}^{m-1}\right) \tag{A.23}$$

$$D_z S_n^m = -S_{n+1}^m \tag{A.24}$$

$$D_{r\times x} S_n^m = -\frac{n+m}{2} S_n^{m-1} + \frac{n-m}{2} S_n^{m+1} \tag{A.25}$$

$$D_{r\times y} S_n^m = -i\frac{n+m}{2} S_n^{m-1} - i\frac{n-m}{2} S_n^{m+1} \tag{A.26}$$

$$D_{r\times z} S_n^m = -im S_n^m \tag{A.27}$$

Substituting into Eq. A.3 and reordering indices, we obtain:

$$v_k = \sum_{n=0}^{\infty} \sum_{m=-n}^{n} v_{kn}^m S_n^m(\vec{r}) \tag{A.28}$$

$$v_{xn}^m = \frac{i}{2}\left(\tilde{\phi}_{n-1}^{m-1} + \tilde{\phi}_{n-1}^{m+1}\right) + \left(-\frac{n+m+1}{2}\chi_n^{m+1} + \frac{n-m+1}{2}\chi_n^{m-1}\right) \tag{A.29}$$

$$v_{yn}^m = \frac{1}{2}\left(\tilde{\phi}_{n-1}^{m-1} - \tilde{\phi}_{n-1}^{m+1}\right) + \left(-i\frac{n+m+1}{2}\chi_n^{m+1} - i\frac{n-m+1}{2}\chi_n^{m-1}\right) \tag{A.30}$$

$$v_{zn}^m = -\tilde{\phi}_{n-1}^m - \chi_n^m im \tag{A.31}$$

We estimate the $\nabla\tilde{\phi}$ error by taking the first neglected term in the truncated expansion:

$$\varepsilon_{\nabla\tilde{\phi},k} \approx \sum_{m=-p}^{p} \left(v_{k,p}^{m}\right)_{\tilde{\phi}} S_p^m(\vec{r}) \tag{A.32}$$

$$\leq \frac{\mathcal{M}_p^{(\tilde{\phi})}}{\tilde{S}_p^0} \frac{(p)!}{r^{p+1}} \tag{A.33}$$

$$\varepsilon_{\nabla\tilde{\phi}} = \sqrt{\varepsilon_{\nabla\tilde{\phi},x}^2 + \varepsilon_{\nabla\tilde{\phi},y}^2 + \varepsilon_{\nabla\tilde{\phi},z}^2} \tag{A.34}$$

$$\lesssim \sqrt{3} \frac{\mathcal{M}_{p-1}}{\tilde{S}_{p-1}^0} \frac{(p)!}{r^{p+1}} \tag{A.35}$$

We estimate the $\nabla \times (\vec{r}\chi)$ error as

$$\varepsilon_{\nabla\times(\vec{r}\chi),k} \approx \sum_{m=-p}^{p} \left(v_{k,p}^{m}\right)_{\chi} S_p^m(\vec{r}) \tag{A.36}$$

$$\leq \frac{\mathcal{M}_p^{(\chi)}}{\tilde{S}_p^0} \frac{(p)!}{r^{p+1}} \tag{A.37}$$

$$\varepsilon_{\nabla\times(\vec{r}\chi)} = \sqrt{\varepsilon_{\nabla\times(\vec{r}\chi),x}^2 + \varepsilon_{\nabla\times(\vec{r}\chi),y}^2 + \varepsilon_{\nabla\times(\vec{r}\chi),z}^2} \tag{A.38}$$

$$\lesssim \sqrt{3} \frac{\mathcal{M}_p^{(\chi)}}{\tilde{S}_p^0} \frac{p!}{r^{p+1}} \tag{A.39}$$

Substituting Eqs. A.32 and A.36 into Eq. A.4, we obtain

$$|\vec{\varepsilon}_S| \lesssim \sqrt{3} \frac{\mathcal{M}_{p-1}}{\tilde{S}_{p-1}^0} \frac{(p)!}{r^{p+1}} + \sqrt{3} \frac{\mathcal{M}_p^{(\chi)}}{\tilde{S}_p^0} \frac{p!}{r^{p+1}} \tag{A.40}$$

which is our estimate of the upper bound of the magnitude of the vector error of a multipole expansion, and the second term can be ignored if the Lamb-Helmholtz decomposition is not used. In summary, we have arrived at multipole and local expansion error upper bounds for the its induced vector field under the Lamb-Helmholtz decomposition for scalar-plus-vector potential fields.

Acknowledgments

The paper was developed based on partial support through the Center for Autonomous Air Mobility and Sensing (CAAMS) under NSF award 2139551, and partial support from NASA under award No. 80NSSC21M0070.

References

[1] B.E. MacNeal, J.R. Brauer, and R.N. Coppolino, A general finite element vector potential formulation of electromagnetics using a time-integrated electric scalar potential, IEEE Transactions on Magnetics, 26(5):1768–1770, 1990.

[2] Luigi Morino, Scalar/vector potential formulation for compressible viscous unsteady flows, National Aeronautics and Space Administration, 1985.

[3] Leslie Greengard and Vladimir Rokhlin, A fast algorithm for particle simulations, Journal of Computational Physics, 73(2):325–348, 1987.

[4] Leslie Greengard and Vladimir Rokhlin, A new version of the fast multipole method for the Laplace equation in three dimensions, Acta Numerica, 6:229–269, 1997.

[5] Barry A. Cipra, The best of the 20th century: Editors name top 10 algorithms, SIAM News, 33(4):1–2, 2000.

[6] Nail A. Gumerov and Ramani Duraiswami, Efficient FMM accelerated vortex methods in three dimensions via the Lamb–Helmholtz decomposition, Journal of Computational Physics, 240:310–328, 2013.

[7] Rio Yokota and L.A. Barba, FMM-based vortex method for simulation of isotropic turbulence on GPUs, compared with a spectral method, Computers & Fluids, 80:17–27, 2013.

[8] Eduardo J. Alvarez and Andrew Ning, Development of a vortex particle code for the modeling of wake interaction in distributed propulsion, 2018 Applied Aerodynamics Conference, pp. 3646, 2018.

[9] Deng Shuai, Chen Jiang, Wang Yunjie, and Wang Haowen, Acceleration of unsteady vortex lattice method via dipole panel fast multipole method, Chinese Journal of Aeronautics, 34(2):265–278, 2021.

[10] Michael S. Warren and John K. Salmon, A portable parallel particle program, Computer Physics Communications, 87(1-2):266–290, 1995.

[11] Walter Dehnen, A hierarchical O(N) force calculation algorithm, Journal of Computational Physics, 179(1):27–42, 2002.

[12] Rio Yokota, An FMM based on dual tree traversal for many-core architectures, Journal of Algorithms & Computational Technology, 7(3):301–324, 2013.

[13] Huda Ibeid, Rio Yokota, Jennifer Pestana, and David Keyes, Fast multipole preconditioners for sparse matrices arising from elliptic equations, Computing and Visualization in Science, 18:213–229, 2018.

[14] Nail A. Gumerov, Shoken Kaneko, and Ramani Duraiswami, Recursive computation of the multipole expansions of layer potential integrals over simplices for efficient fast multipole accelerated boundary elements, Journal of Computational Physics, 486:112118, 2023.

[15] Michael A. Epton and Benjamin Dembart, Multipole translation theory for the three-dimensional Laplace and Helmholtz equations, SIAM Journal on Scientific Computing, 16(4):865–897, 1995.

[16] Nail A. Gumerov and Ramani Duraiswami, Comparison of the efficiency of translation operators used in the fast multipole method for the 3D Laplace equation, UMIACS Technical Report, 2011.

[17] Nail A. Gumerov and Ramani Duraiswami, Recursive computation of spherical harmonic rotation coefficients of large degree, arXiv preprint arXiv:1403.7698, 2014.

[18] Holger Dachsel, Corrected Article: "An error-controlled fast multipole method" [J. Chem. Phys. 131, 244102 (2009)], The Journal of Chemical Physics, 132(11), 2010.

[19] Gavin J. Pringle, Numerical study of three-dimensional flow using fast parallel particle algorithms, PhD thesis, Napier University of Edinburgh, 1994.

[20] Dorth Sølvason and Henrik G. Petersen, Error estimates for the fast multipole method, Journal of Statistical Physics, 86:391–420, 1997.

[21] Horace Lamb, Hydrodynamics, University Press, 1924.

[22] Eduardo J. Alvarez, Judd Mehr, and Andrew Ning, FLOWUnsteady: an interactional aerodynamics solver for multirotor aircraft and wind energy, AIAA Aviation 2022 Forum, pp. 3218, 2022.

[23] Eduardo J. Alvarez and Andrew Ning, Stable vortex particle method formulation for meshless large-eddy simulation, AIAA Journal, 62(2):637–656, 2024.

[24] J. Kurzak and B.M. Pettitt, Fast multipole methods for particle dynamics, Molecular Simulation, 32(10-11):775–790, 2006.

[25] Holger Dachsel, Michael Hofmann, Jens Lang, and Gudula Rünger, Automatic tuning of the fast multipole method based on integrated performance prediction, 2012 IEEE 14th International Conference on High Performance Computing and Communication & 2012 IEEE 9th International Conference on Embedded Software and Systems, pp. 617–624, 2012.

[26] Rio Yokota and L.A. Barba, Parameter tuning of a hybrid treecode-fmm on gpus, The First International Workshop on Characterizing Applications for Heterogeneous Exascale Systems. Tuscon, Arizona: hgpu.org, 2012.

[27] Ross Adelman, Nail A. Gumerov, and Ramani Duraiswami, FMM/GPU-accelerated boundary element method for computational magnetics and electrostatics, IEEE Transactions on Magnetics, 53(12):1–11, 2017.

[28] Nail A. Gumerov and Ramani Duraiswami, Fast multipole methods on graphics processors, Journal of Computational Physics, 227(18):8290–8313, 2008.

[29] Hongwei Cheng, Leslie Greengard, and Vladimir Rokhlin, A fast adaptive multipole algorithm in three dimensions, Journal of Computational Physics, 155(2):468–498, 1999.

[30] Walter Dehnen, A fast multipole method for stellar dynamics, Computational Astrophysics and Cosmology, 1:1–23, 2014.

[31] Ivo Kabadshow, Periodic boundary conditions and the error-controlled fast multipole method, Forschungszentrum Jülich, 11, 2012.

[32] Leslie Greengard, The rapid evaluation of potential fields in particle systems, MIT Press, 1988.

[33] Leslie Greengard, Michael O'Neil, Manas Rachh, and Felipe Vico, Fast multipole methods for the evaluation of layer potentials with locally-corrected quadratures, Journal of Computational Physics: X, 10:100092, 2021.

[34] Viktor Pabyrivskyi, Petro Pukach, Nelya Pabyrivska, and Galyna Beregova, Concretization of the mathematical models of linear elasticity theory by representation of general solution in terms of harmonic potentials in Papkovich-Neuber form, Proceedings of the Romanian Academy Series A–Mathematics, Physics, Technical Sciences, Information Science, 23(2):113–121, 2022.

[35] Rio Yokota and Lorena Barba, Hierarchical n-body simulations with autotuning for heterogeneous systems, Computing in Science & Engineering, 14(3):30–39, 2012.

[36] Mattia Barbarino and Davide Bianco, A BEM–FMM approach applied to the combined convected Helmholtz integral formulation for the solution of aeroacoustic problems, Computer Methods in Applied Mechanics and Engineering, 342:585–603, 2018.

[37] Jeff Bezanson, Alan Edelman, Stefan Karpinski, and Viral B. Shah, Julia: A fresh approach to numerical computing, SIAM Review, 59(1):65–98, 2017.

[38] Christopher A. White and Martin Head-Gordon, Rotating around the quartic angular momentum barrier in fast multipole method calculations, The Journal of Chemical Physics, 105(12):5061–5067, 1996.

[39] Samer Salloum and Issam Lakkis, An adaptive error-controlled hybrid fast solver for regularized vortex methods, Journal of Computational Physics, 468:111504, 2022.

[40] John L. Hess and A.M.O. Smith, Calculation of potential flow about arbitrary bodies, Progress in Aerospace Sciences, 8:1–138, 1967.

[41] Lothar Birk, A comprehensive and practical guide to the Hess and Smith constant source and dipole panel, Ship Technology Research, 69(1):50–62, 2022.