Fast Multipole Accelerated Unsteady Vortex Lattice Method Based Computations

Abu Kebbie-Anthony,∗ Nail A. Gumerov,‡ Sergio Preidikman,§ Balakumar Balachandran,¶ and Shapour Azarm§

University of Maryland, College Park, Maryland 20742

DOI: 10.2514/1.I010690

The authors present an accelerated aerodynamic computational model derived from the integration of the fast multipole method (FMM) with the unsteady vortex lattice method (UVLM) based aerodynamic model. It is determined that the performance of this computational model depends on the tuning of some FMM parameters and that there is a tradeoff between the computational speed and the accuracy of the computed loads. This tradeoff is examined by varying the truncation number, the order of the Gauss–Legendre quadrature, and the clustering parameter for the wake velocity calculations. Results of the computational cost reduction study achieved through the accelerated aerodynamic simulator are reported for a planar, rectangular lifting surface with a high aspect ratio. The computational approach presented in this paper is the first in the literature wherein the FMM has been implemented for an UVLM-based nonlinear, unsteady aerodynamic simulator. The approach has broad applicability for the study of aerodynamic and aeroelastic responses of aircraft systems and related decision support systems in dynamic data-driven application systems.

Nomenclature

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A(L, r)$</td>
<td>velocity field kernel corresponding to vortex segment $L$ and field point $r$</td>
</tr>
<tr>
<td>$A^h(r, t)$</td>
<td>vector potential of fluid particle (field point) $r$ due to far-field interactions at time $t$ for each Cartesian component ($h = 1, 2, 3$)</td>
</tr>
<tr>
<td>$c_{nm}$</td>
<td>expansion coefficients for vector potential</td>
</tr>
<tr>
<td>$c_D$</td>
<td>drag coefficient for lifting surface</td>
</tr>
<tr>
<td>$c_{D_i}$</td>
<td>induced drag coefficient for lifting surface</td>
</tr>
<tr>
<td>$c_L$</td>
<td>lift coefficient for lifting surface</td>
</tr>
<tr>
<td>$c_N$</td>
<td>normal force coefficient for lifting surface</td>
</tr>
<tr>
<td>$\partial_i$</td>
<td>partial time derivative</td>
</tr>
<tr>
<td>$D_h^{(lm)}$</td>
<td>local expansion coefficients for each Cartesian component ($h = 1, 2, 3$)</td>
</tr>
<tr>
<td>$d_n$</td>
<td>size of box in $n$th level of octree subdivision</td>
</tr>
<tr>
<td>$e_1, e_2, e_3$</td>
<td>unit vectors of $r_{1i}$ and $r_{2i}$, respectively</td>
</tr>
<tr>
<td>$G(r, r')$</td>
<td>multipole expansion of a monopole source located at $r'$ for field point $r$</td>
</tr>
<tr>
<td>$G(t)$</td>
<td>circulation of vortex ring at time $t$</td>
</tr>
<tr>
<td>$K$</td>
<td>total number of time steps</td>
</tr>
<tr>
<td>$k$</td>
<td>time step index</td>
</tr>
<tr>
<td>$L$</td>
<td>vector along vortex segment</td>
</tr>
<tr>
<td>$l_i$</td>
<td>level in the octree</td>
</tr>
<tr>
<td>$l_r$</td>
<td>unit vector of $L_r$</td>
</tr>
<tr>
<td>$M$</td>
<td>total number of field points (receivers)</td>
</tr>
<tr>
<td>$N$</td>
<td>total number of vortex segments (sources)</td>
</tr>
<tr>
<td>$N_p$</td>
<td>size of relatively small body meshes</td>
</tr>
<tr>
<td>$N_q$</td>
<td>order of Gauss–Legendre quadrature</td>
</tr>
<tr>
<td>$N_{(R)m}$</td>
<td>normalization coefficient for regular spherical basis function</td>
</tr>
<tr>
<td>$N_{(S)m}$</td>
<td>normalization coefficient for singular spherical basis function</td>
</tr>
<tr>
<td>$\hat{n}$</td>
<td>unit vector normal to surface</td>
</tr>
<tr>
<td>$p(r, t)$</td>
<td>pressure associated with fluid particle (field point) $r$ at time $t$</td>
</tr>
<tr>
<td>$p_F$</td>
<td>truncation number of fast multipole method algorithm</td>
</tr>
<tr>
<td>$r$</td>
<td>position vector to fluid particle</td>
</tr>
<tr>
<td>$R_m(r)$</td>
<td>regular spherical basis function of field point $r$</td>
</tr>
<tr>
<td>$\text{res}(-)$</td>
<td>residual function</td>
</tr>
<tr>
<td>$r_c$</td>
<td>center of receiver box</td>
</tr>
<tr>
<td>$r_{1i}, r_{2i}$</td>
<td>position vectors from endpoints of vortex segment to field point $r$</td>
</tr>
<tr>
<td>$(r, \theta, \phi)$</td>
<td>spherical coordinates of field point $r$</td>
</tr>
<tr>
<td>$S_m^m(r)$</td>
<td>singular spherical basis function of field point $r$</td>
</tr>
</tbody>
</table>
I. Introduction

Due to unforeseen system responses and environmental conditions, the mission success of unmanned aircraft systems (UASs) can be adversely influenced. These unexpected events make the UAS quite susceptible to aeroelastic instabilities. An UAS subjected to aeroelastic instabilities long enough will fail, depending on the material properties and design of the aircraft, as well as on the type of instability. Current designs for surveillance and observation platforms and data collection drones, such as the joined-wing SensorCraft [1, 2], have high-aspect-ratio wings for efficiency. The increased aspect ratio results in high flexibility and brings forth the necessity for consideration of aeroelastic loads in design. Advanced surveillance and observation platforms and data collection drones, such as the joined-wing SensorCraft [1, 2], have high-aspect-ratio wings for efficiency. The increased aspect ratio results in high flexibility and brings forth the necessity for consideration of aeroelastic loads in design. As reported in this group’s prior work, systems with flexible wings can exhibit complex motions [3]. These motions cannot be sufficiently captured by a weak coupling of structural dynamics and aerodynamics. Linear aeroelastic models, although amenable to fast computations, are not adequate to capture postcritical aeroelastic behavior, and these models cannot be used to estimate reliable margins for aeroelastic instabilities; that is, to generate a safe maneuvering envelope. In this regard, there is an urgent need to develop a set of fast, robust, accurate, and reliable prediction methods based on fully coupled aerodynamics, structural dynamics, control systems, and nonlinear analysis.

This task of predicting and mitigating nonlinear aeroelastic effects can be accomplished by a decision support system (DSS). A DSS could be developed through the combining of nonlinear aeroelastic simulations with measurement data through the dynamic data-driven application systems (DDDAS) paradigm. This paradigm can be used to realize a framework in which measurement data, such as those obtained from sensors, are collected for a physical system and used to dynamically update a computational model of it. Subsequently, the computational model will be used to steer the measurement process. Darema [4] introduced the DDDAS concept in 2004. In the studies of Farhat and Amsallem [5] and Allaire et al. [6], the DDDAS paradigm has been used to predict the failure and degradation in UASs and tailor mission plans to best suit the remaining capabilities of the aircraft. With the DDDAS paradigm, there is an emphasis on the speedup of simulations. Simulations can be performed in advance of a mission in order to predict likely scenarios. Fast computations mean more scenarios can be sampled, thus providing a more complete picture of the mission. After realization of scenarios, the speed of the calculations can help improve the speed at which the model can be updated. Given the variety of scenarios that arise due to unforeseen events in the system and environment, more simulation data can be used to predict the expected response of the system. Furthermore, metamodels can be developed to approximate the nonlinear models, thereby helping reduce the number of execution calls of the full aeroelastic simulator for decision support [7–9].

The aeroelastic model [10, 11] is developed through the integration of a structural dynamics model, a finite element (FE) model, an acoustic model, the unsteady vortex lattice method (UVLM), and a procedure for the intermodel connection: that is, for transferring information between the aerodynamic simulator and the structural dynamics simulator. The bidirectional exchange of information between the simulators for aerodynamics and structural dynamics is a part of a co-simulation strategy [11–13].

In this paper, the authors focus on accelerating the calculations involved with computing the aerodynamic loads by algorithmically reducing the complexity. The major source of computational complexity in the aerodynamic model used for this work lies in the computation of the wake velocities. The high computational cost can be observed in the aeroelastic simulation results obtained for a representative joined-wing SensorCraft and shown in Fig. 1. In Fig. 2, it is shown that over 90% of the computation time involved in the aeroelastic simulation of the joined-wing SensorCraft is spent on evaluating the wake velocity over the large domain size. The computational time required to compute the structural response (that is, to numerically integrate the structure’s equations of motion) is neglected in the figure because this time is dwarfed by the unsteady aerodynamic calculations. For similar reasons, the processes of the aerodynamic model involving the evaluation of the distribution of circulation and the calculations of the aerodynamic loads are negligible. This observation emphasizes the need for acceleration of the free deforming wake velocity computations in the aerodynamic model to accelerate the entire aeroelastic simulation.

![Fig. 1 Mesh of representative joined-wing SensorCraft.](image-url)
For the aerodynamic simulator, the UVLM-based scheme [10,11,14] is used to compute the flowfield around a body and the forces acting on it. This method can be used to model fluid flow over three-dimensional lifting and nonlifting surfaces (such as the wings and fuselage of the aircraft, respectively) undergoing arbitrary time-dependent deformations and any motion in space. Unlike the FE methods for which one creates a mesh, in an UVLM scheme, one uses the discrete points on the surface, making it more amenable to dynamically moving structures. Bound-vortex sheets, which model the fluid boundary layers, are replaced by a lattice of short, straight vortex segments with time-varying circulations $\Gamma(t)$. These segments are used to divide the body's surface into several elements of area (the so-called panels that are generally nonplanar). The model is completed by joining free-vortex lines, representing the wakes to be shed from the sharp edges of the wing, to the bound-vortex lattice along the separation edges, such as the wing trailing edges and the wingtips. The locations at which separation occurs are provided as inputs. However, in the wake, the positions of the vortex segments and their circulations are determined by the motion of the surface. As time is evolved, the number of vortex segments associated with the wake grows in proportion to the number of vortex segments on the trailing edge and wingtips.

To calculate the velocity field by using the UVLM, based on the Biot–Savart law [14], the influence of $N$ discrete finite vortex segments with spatially constant/time-varying circulation must be computed; the associated computational cost is of $O(N^2)$. It should be noted that, due to the convection of the wake, the value of $N$ increases as time increases. To mitigate this computational cost, Chabalko et al. [15] and Chabalko and Balachandran [16,17] sought to distribute two-dimensional vortex interaction calculations over the cores of a graphics processing unit (GPU). In the development of the aerelastic computational framework called Flexit, Fleischmann et al. [18] used GPU computing to attain fast simulations for the UVLM aerodynamic model. Although effective in dispersing the workload over more units, the speedup gained from GPU computing is hardware dependent, which puts a cap on how well this framework scales with large-sized problems. By comparison, an algorithmic approach is explored here.

The fast multipole method (FMM) has been identified as one of the 10 algorithms with the greatest influence on the development and practice of science and engineering in the 20th century [19]. With the FMM, the computational cost of the $N$-body problem is reduced from $O(N^2)$ to $O(N \log N)$, which is essential for any practical application when $N$ becomes large. The FMM algorithm was first developed in 1987 by Greengard and Roklin [20] to calculate gravitational and electrostatic potentials. This algorithm was further improved by Carrier et al. [21] for the evaluation of potential and force fields in systems involving large numbers of particles. In the work reported in Refs. [22,23], the FMM algorithm was applied toward the evaluation of Laplace’s equation governing fluid flow. The FMM has been shown to be helpful with approximating fluid flows governed by the Navier–Stokes equations through vortex methods. Gumerov and Duraiswami [24] implemented the FMM algorithm for simulations based on vortex methods via the Lamb–Helmholtz decomposition. Recently, application of the FMM algorithm with a discrete vortex method for free domain and periodic problems has been presented by Ricciardi et al. [25]. Cheung et al. [26] applied an octree data structure similar to that used in the FMM to speed up the calculation of wake velocities. The acceleration gained through the application of the FMM can be further enhanced through hardware usage, such as GPUs [27–29].

Although in the literature [25] it has been reported that the use of the FMM algorithm can be used to computationally accelerate aerodynamic simulations, an examination of the acceleration of the UVLM scheme by using the FMM has not been carried out thus far. This is addressed in the current work, wherein the authors carry out the following:

1) Implement the FMM for accelerating the UVLM-based aerodynamic simulator.
2) Study the performance and accuracy tradeoff of the FMM by tuning the parameters (i.e., truncation number, order of quadrature, and clustering parameter/number of levels in octree) for the FMM accelerated UVLM-based aerodynamic simulator.
3) Report the computational cost reduction of the FMM accelerated UVLM for a planar, rectangular lifting surface with varying aspect ratios. This paper is an extension of the previous study by the authors [30], in which the computational cost reduction achieved from the accelerated UVLM was presented.

The rest of the paper is organized as follows: In Sec. II, the aerodynamic model used in this work is discussed. A general description of the FMM and its performance/accuracy tradeoff parameters are discussed in Sec. III. To illustrate the FMM accelerated UVLM capabilities, a numerical study for a planar, rectangular lifting surface is conducted with varying levels of refinement for the mesh and FMM settings. These results are presented in Sec. IV. Finally, concluding remarks and thoughts on future work are collected together in Sec. V.

II. Aerodynamic Model

Aerelastic computations are carried out by using a co-simulation strategy (e.g., Ref. [11]). For this work, in the aerelastic simulator, the authors use the UVLM to predict the aerodynamic loads on the lifting surfaces. Additionally, this solver is coupled with a structural dynamics simulator to capture dynamic aerelasticity. Co-simulation here refers to subdivision of a system with coupled physics into subsystems that are simultaneously simulated and numerically combined with a suitable exchange of states at predefined time instances to account for the strong coupling. Co-simulations can consist of any number of subsystems but, in this effort, two subsystems are included in the co-simulation strategy.
The first subsystem is the UAS structural model, which is obtained through the use of the FE method. The second subsystem is the UAS aerodynamic model, which is obtained on the basis of the UVLM.

A. Unsteady Vortex Lattice Method

In this work, the UVLM is used to compute aerodynamic loads. The UVLM is a surface vorticity model that is used to accurately approximate the physics for a very large-Reynolds-number fully attached flow. The infinitesimally thin layers of vorticity may be viewed as an infinite Reynolds number approximation to the actual boundary layers. The UVLM can be applied to lifting surfaces of any planform, camber, and twist; the lifting surfaces may undergo any time-dependent deformation and execute any maneuver in moving air. The flow surrounding the lifting surface is assumed to be inviscid, incompressible, and irrotational over the entire flowfield, except at the solid boundaries and in the wakes. Due to the relative motion between the wing and the fluid, as well as the viscous effects, vorticity is generated in the fluid in a thin region next to the wing’s surface (the boundary layer). The boundary layers on the upper and lower surfaces are merged into a single vortex sheet.

The bound-vortex sheets are replaced by lattices of short, straight vortex segments with spatially constant/time-varying circulation $\Gamma(t)$. These segments divide the wing surface into a finite number of typically nonplanar, quadrilateral elements of area with straight edges called panels (or vortex rings). The model is completed by joining the free-vortex lattices (wakes) to the bound-vortex lattice (lifting surface) along the separation edges, such as the trailing edges and leading edges of the lifting surface. The separation locations are user supplied. Each vortex ring has a single unknown circulation $G(t)$ instead of the four unknown circulations around each of the short vortex-line segments along its edges. Consequently, the requirement of spatial conservation of circulation is automatically satisfied throughout the lattices. Once the values of $G(t)$ for each panel are known, the $\Gamma(t)$ all of the straight vortex segments can be conveniently determined. The governing equation is complemented with the following boundary conditions:

1) Regularity at infinity: This condition requires that the velocity field associated with the disturbance decays away from the body and its wake. Hence,

$$\lim_{|r| \to \infty} \|V_B(r, t) + V_W(r, t)\|_2 = \|V_\infty\|_2$$  

where $V_B(r, t)$ and $V_W(r, t)$ are the velocity fields associated with the vorticity in the boundary layers on the body surface and the vorticity in the free vortex shed from the wing’s trailing edge (including the tip), respectively; and $V_\infty$ is the freestream velocity. The velocity field obtained from the Biot–Savart law identically satisfies this condition.

2) No-penetration condition: This condition requires that, at every point of the solid surface, the normal component of the fluid velocity relative to the body’s surface must be zero:

$$(V_B(r, t) + V_W(r, t) + V_\infty - V_{\text{Body}}) \cdot \hat{n} = 0$$

where $V_{\text{Body}}$ is the velocity of the body’s surface, and $\hat{n}$ is a unit vector normal to the surface. Equation (2) is only imposed at the control point located in the geometric center of each panel. To satisfy the unsteady Kutta condition at each time step, the vortex rings along the edges are shed into the flow, where they have the same order as they had on the wing's surface. The vortex rings are moved downstream with the flow by moving the endpoints of their vortex segments, called nodes, with the local fluid–particle velocity $V$ to new positions, denoted $r(t + \Delta t)$, according to the first-order approximation:

$$r(t + \Delta t) \approx r(t) + V(r, t)\Delta t$$

B. Aerodynamic Loads

The aerodynamic loads acting on the lifting surface are computed as follows: for each element of the bound lattice, the force is determined based on the pressure jump across the lifting surface at the control point. This calculation is carried out by using the unsteady Bernoulli equation:

$$\partial_t \phi(r, t) + \frac{1}{2} V(r, t) \cdot V(r, t) + \frac{p(r, t)}{\rho} = W(t)$$

where $\partial_t$ denotes the partial time derivative at a fixed location in an inertial reference frame, $V(r, t)$ is the spatial gradient of the scalar velocity potential $\phi(r, t)$, $\rho$ is the fluid density, $p(r, t)$ is the pressure, and $W(t)$ is the total energy per unit mass, which only depends on time and has the same value at every point in the domain of the flow.

C. Formulation

In Fig. 2, the authors show a typical vortex ring (where the circulation of the individual vortex segments is the same as the circulation of the vortex ring) and how these vortex segments contribute to the velocity at the field point $r$. The Bio–Savart integral can be transformed as follows:

$$V_i(r, t) = \frac{\Gamma_i(t)}{4\pi} \int_{L_i} \frac{dr' \times (r - r')}{|r - r'|^2} = \Gamma_i(t) \nabla \times \int_{L_i} G(r, r') \, dr'$$

where $G(r, r')$ is the multipole expansion of a monopole source located at point $r'$. The velocity field satisfying Eq. (5) at the field point $r$ for time $t$ $V_i(r, t)$ associated with a discrete segment of a straight line vortex $L_i$, $i = 1, 2, \ldots, N$ of circulation strength $\Gamma_i(t)$ can be evaluated as follows:

$$V_i(r, t) = \frac{\Gamma_i(t)}{4\pi} \frac{L_i \times r_{i1}}{|L_i \times r_{i1}|^2} \left[ L_i \cdot (\hat{e}_{i1} - \hat{e}_{i2}) \right] \equiv A_i(L_i, r) \Gamma_i(t)$$

Here, $r_{i1}$ and $r_{i2}$ are the position vectors from the endpoints of the vortex segment $L_i = r_{i1} - r_{i2}$ to the field point $r$, and $\hat{e}_{i1}$ and $\hat{e}_{i2}$ are unit vectors in the directions of $r_{i1}$ and $r_{i2}$, respectively. To avoid the singularity that appears when the point approaches the vortex line or its extension, the term $\delta |L_i|^2$ can be introduced to Eq. (2) to obtain
Biot–Savart error, there is good agreement between the aerodynamic loads computed from Eqs. (6) and (7). The lift coefficient
1 m with nine panels in the chordwise direction and a wing span of 4 m with 36 panels in the spanwise direction. Given the small magnitude of the

cDi
performed directly, whereas the dense matrix-vector product is approximated via the use of data structures, the generation of multipole expansions, a n d

velocity field calculations. The FMM can be used to reduce the computational cost from the scale of a full aircraft, these computations become intractable. Such computational expense motivates the need for the FMM to accelerate the

The computational cost of the simulation increases significantly with the number of field points and vortex segments. As time evolves, for a system on

The influence of the cutoff radius δ, or smoothing parameter, on the velocity is strongly felt in the immediate vicinity of the vortex line but is hardly noticeable elsewhere. In Fig. 4, the relative errors of the computed aerodynamic load coefficients obtained by using Eqs. (6) and (7) are plotted. The corresponding simulations have been carried out for a planar, rectangular wing with an aspect ratio of four and subjected to a

freestream velocity with a magnitude of 125 m/s, an angle of attack of 5 deg, and an air density of 1.255 kg/m³. The wing has a chord length of 1 m with nine panels in the chordwise direction and a wing span of 4 m with 36 panels in the spanwise direction. Given the small magnitude of the error, there is good agreement between the aerodynamic loads computed from Eqs. (6) and (7). The lift coefficient cL, and induced drag coefficient cDi obtained with the UVLM-based scheme were compared against those obtained from Prandtl’s lifting line theory to validate the UVLM-based aerodynamic model. The results obtained for the lift and induced drag coefficients from the UVLM scheme were found to be in good agreement with the values by using Prandtl’s lifting line theory.

The velocity field at point r can be computed as the summation of velocity fields associated with each of the discrete vortex segments

L1, L2, . . . , LN at the field point r:

V(r, t) = \sum_{i=1}^{N} V_i(r, t) = \sum_{i=1}^{N} A_i(L_i, r) \Gamma_i(t) (8)

At each time step, the velocity field needs to be evaluated at M field points, rj, j = 1, 2, . . . , M, which leads to a computational cost of O(NM).

The computational cost of the simulation increases significantly with the number of field points and vortex segments. As time evolves, for a system on

the scale of a full aircraft, these computations become intractable. Such computational expense motivates the need for the FMM to accelerate the velocity field calculations. The FMM can be used to reduce the computational cost from O(NM) to O(N + M), making the runtime more practical.

III. Fast Multipole Method

The FMM is a hierarchical algorithm, which can be used to speed up matrix-vector products. The main idea of the FMM is to split system equation (8) into near-field and far-field interactions. This is done through the decomposition of the dense matrix into sparse and dense parts as follows:

V(rj, t) = \sum_{i=1}^{N} A(L_i, rj) \Gamma_i(t) = \sum_{i=1}^{N} A^{(sp))}(L_i, rj) \Gamma_i(t) + \sum_{i=1}^{N} A^{(dme)}(L_i, rj) \Gamma_i(t), \quad j = 1, 2, . . . , M (9)

where L1, L2, . . . , LN are the sources (vortex segments) and r1, r2, . . . , rM are the receivers (field points). The sparse matrix-vector product is performed directly, whereas the dense matrix-vector product is approximated via the use of data structures, the generation of multipole expansions, and the evaluation of local expansions. This means that the interactions between near-field pairs of field points and vortex segments are directly computed,
whereas the interactions between the far-field pair of field points and vortex segment pairs are approximated. More details on the basics of the FMM can be found elsewhere [19–24,27,31]. Here, the authors briefly describe the main components of the algorithm and the specifics for vortex filament computations and use in the UVLM.

### A. Data Structure

The hierarchical data structure (usually octree for three-dimensional simulations) of the FMM serves two purposes. First, it is needed for a fast neighbor search to compute the near-field interactions directly by using $O(N)$ or $O(N \log N)$ operations. Second, it is needed to organize far-field interactions in a hierarchical way, which can be done with the same computational complexity as the near-field interactions. The entire computational domain is enclosed in a cube, or box, of size $d_0 \times d_0 \times d_0$, which is said to be subdivision level 0. This cube is partitioned into eight equal cubes of size $d_1 = d_0/2$, which form subdivision level 1. This process of partitioning the volume continues until the maximum number of source points (the centers of the vortex segments) in a box does not exceed some number $s$ called the “clustering parameter.” The value of $s$ depends on several parameters and is a subject for tuning; see the following. This level of subdivision is the maximum level in the octree, denoted as $l_{\text{max}}$. So, at this level, $8^{l_{\text{max}}}$ cubes of size $d_{l_{\text{max}}} = d_0/2^{l_{\text{max}}}$ are used to partition the entire computational domain. In the UVLM-based scheme, the sources are located on a subset of surfaces and most of the cubes in such a tree are empty. Because the FMM is an adaptive algorithm, one skips all empty boxes, and the actual number of cubes at some level $l$ is much smaller than $8^l$. There are variations of the FMM formulation in which one uses data trees (in this case, the “leaves” of the “tree” can be located at any level not exceeding $l_{\text{max}}$, for example, Ref. [22]); this can be called “fully adaptive” FMM and data pyramids (all leaves of the tree are located at level $l_{\text{max}}$, for example, Ref. [27]). Details and practical issues on efficient implementation of data structures and comparisons of “adaptive” and fully adaptive FMM variations can be found in Ref. [31]. It is noticeable that computations usually related to data structures do not exceed 10% of the complexity of the entire FMM. This is why the authors have used standard procedures (bit interleaving and sorting to form child, parent, and neighbor lists) in the present study. However, substantially nonuniform accelerations of the algorithm can be achieved by using additional GPU accelerations; in which case, the methods for computing data structures should be significantly modified [30].

### B. Multipoles

The velocity field generated by a vortex segment is not a potential, but it can be expressed in terms of three dependent scalar harmonic functions or two independent scalar harmonic functions by using the Lamb–Helmholtz decomposition [24]. The FMM for harmonic functions is well developed and studied. The far field, or multipole expansion of a monopole source located at point $r'$ with respect to the center $r_c$ of the cube containing the source, can be represented in the form of a series:

$$G(r, r') = \frac{1}{4\pi ||r - r'||_2} = \sum_{p=0}^{p_{\text{max}}} \sum_{n=-n}^{n} R_{n}^{m}(r_c - r') S_{n}^{m}(r - r_c) + O \left( \left( \frac{||r - r'||_2}{||r - r'|'} \right)^{p_{\text{max}}} \right)$$

(10)

where $r$ is a field point, such that $||r - r'||_2 > ||r' - r_c||_2$, $p_{\text{max}}$ is the FMM truncation number, whereas $R_{n}^{m}$ and $S_{n}^{m}$ are regular and singular spherical basis functions that, generally, can be written as follows:

$$R_{n}^{m}(r) = r^n Y_{n}^{m}(\theta, \phi),$$

$$S_{n}^{m}(r) = r^n N_{n}^{(R)m}(\theta, \phi)$$

(11)

Here, $(r, \theta, \phi)$ are the spherical coordinates of point $r$, $Y_{n}^{m}(\theta, \phi)$ are the spherical harmonics, and $N_{n}^{(R)m}(\theta, \phi)$ are normalization coefficients. There are different normalizations of spherical harmonics in the literature. Also, real or complex harmonics can be used (see Refs. [23,27]).

Consider the field of vortex segments specified by a unit vector $I_i = -L_i/||L_i||_2$, center $r_i = (r_{i1} + r_{i2})/2$, and circulation $\Gamma_i(t)$. Bio–Savart integral equation (5) can be transformed as follows:

$$V_i(r, t) = \nabla \times \left( \frac{\Gamma_i(t)}{2} I_i \int_{-1}^{1} G \left( r, r_i - \frac{1}{2} I_i \xi \right) \, d\xi \right) = \nabla \times A_i(L_i, r) \Gamma_i(t)$$

(12)

These expressions show several facts important for the FMM implementation. First, sums of the fields of vortex segments can be represented as a curl of sums of the respective vector potentials $A_i(L_i, r)$. This means that the FMM can be applied to the summation of vector potentials and then the curl of the obtained field can be computed. Second, each Cartesian component of $A_i(L_i, r)$ is a scalar harmonic function. This means that the FMM for the scalar Laplace equation can be applied to each component independently. Third, Eq. (10) can be used to obtain the multipole expansion for the vector potential of each vortex segment:

$$A_i(L_i, r) = \frac{\Gamma_i(t)}{2} \sum_{n=0}^{n_{\text{max}}} \sum_{m=-n}^{n} C_{(0)n}^{m}(r - r_i) + O \left( \max(||r_{i1} - r_c||_2, ||r_{i2} - r_c||_2) \right)$$

(13)

Expansion coefficients $C_{(0)n}^{m}$ can be calculated by using the Gauss–Legendre quadrature of order $N_q$ with weights $w_{q, a}$, abscissas $\xi_{q, a}$, and residual $\text{res}(N_q)$:

$$C_{(0)n}^{m} = \sum_{a=1}^{N_q} w_{q, a} R_{q}^{m} \left( r_c - r_i + \frac{1}{2} I_a \xi_{q, a} \right) + \text{res}(N_q)$$

(14)

It is noted that functions

$$R_{q}^{m} \left( r_c - r_i + \frac{1}{2} I_a \xi_{q, a} \right)$$
are polynomials of degree $n$ of $\xi_i$. Hence, Gauss–Legendre quadrature equation (14) provides an exact result for $N_q > n/2$. According to Eq. (13), $n \leq p_F - 1$. Hence, selection

$$N_q = \left\lfloor \frac{p_F - 1}{2} \right\rfloor + 1$$

(15)
guarantees $\text{res}(N_q) = 0$: that is, zero quadrature error in the FMM. However, this requirement is not necessary and may be relaxed; as for the overall accuracy of the method, it is sufficient to balance residuals in Eqs. (13) and (14). In many cases, selection $N_q = 1$ or $N_q = 2$ yields good results.

C. Use of Standard FMMs for the Laplace Equation

After the multipole expansions are computed for each source box at level $l_{\text{max}}$, the rest of the FMM procedure is standard and is along the lines of what is available in the literature (e.g., Refs. [22, 27]). It consists of multipole-to-multipole translation in the upward pass (which recursively produces the multipole expansions for all source boxes at levels $l_{\text{max}}$, $l_{\text{max}} - 1$, . . . , 2) and multipole-to-local and local-to-local translations in the downward pass (which recursively produces local expansions for all receiver boxes at levels 2, 3 . . . , $l_{\text{max}}$). Finally, for a receiver box centered at $r_r$, the authors obtained the values of local expansion coefficients $D^{(h)\text{lm}}_n$ for each Cartesian component ($h = 1, 2, 3$) of the vector potential due to far-field interactions:

$$A^h(r, t) = \sum_{n=0}^{p_F-1} \sum_{m=-n}^{n} D^{(h)\text{lm}}_n \phi^{(h)\text{lm}}(r - r_r)$$

(16)

To complete computation of the dense matrix–vector product, one needs to compute the curl of the vector potential. This can be done by computing the gradient of each component of the vector potential and combining the results into a single vector. The gradient can be computed by using analytical expressions for the derivatives of functions $\phi^{(h)\text{lm}}(r)$.

In this context, the authors note that, currently, a number of open-source and commercial FMM software packages are available. They can be used as “black-box” solvers, which are not the most computationally efficient but are simple ways to obtain acceleration of the UVLM. Indeed, according to Eqs. (10), (13), and (14), the far-field approximation of the $l_{th}$ Cartesian component of the vector potential is

$$A^h(r, t) = \frac{\Gamma(t)}{2} \sum_{j=1}^{N_q} w_j G \left( r, r_j - \frac{1}{2} l_j \xi_j \right)$$

(17)

This means that each vortex segment is represented by $N_q$ monopole sources and that the case should be set for $NN_q$ monopole sources of intensity $(1/2)\Gamma(t)l_j^2 w_j$ located at $r_j - (1/2) l_j \xi_j$ and $M$ receivers. The scalar FMM routine should be called three times (for each Cartesian component, $h = 1, 2, 3$). The result should be obtained in the form of gradients (which is standard for the FMMs computing forces), from which the curl of the vector potential can be formed. An additional user procedure should be created out of the core of the standard FMM (if the standard FMM does not allow customized near-field interactions). With this procedure, one should be able to compute the near-field interactions and subtract from them the near field of the standard FMM.

In the current study, the authors used a specialized FMM code, which is more efficient than the black-box approach because it allows the amortizing and vectorizing of a number of operations. The authors also compared two versions of the FMM: one as described previously (for three scalar harmonic functions); and the other with two scalar harmonic functions, as described in Ref. [24]. The results of a comparison of these versions optimized, as described in the following section, are shown in Fig. 5. The wall-clock time was measured on a 32-core PC (Intel Xeon E5-2683 with 2.1 GHz and 128 GB of RAM). When using the same truncation number $p_F$ for harmonic functions in both methods, the method from Ref. [24] is found to be about 20% faster. However, with the method from Ref. [24], larger errors are produced. As a result, the overall performance in terms of speed versus error for the present method and the method from Ref. [24] is about the same. In the case when relative errors of the order $<10^{-6}$ are acceptable, the present method can be preferable and simpler for implementation because for the method discussed in Ref. [24]; one requires special translation operators, which are not found in the standard FMM.

Fig. 5 Comparison of computation time versus the error for two versions of the FMM: present work and previous work [24]. Computations are performed for all pairwise interactions of $N = 1,048,576$ vortex filaments randomly distributed inside a cubic domain. The relative $L_2$-norm error was measured by using 1000 random control points at which solutions were computed without the use of the FMM.
The wall-clock times reported in the following were measured on an Intel® Xeon® CPU E3-1245 v5 (3.50 GHz) eight-core PC with 16 GB of RAM. Table 1. Therefore, the FMM parameters found in this study are general and can be applied to cases with various flow conditions and geometries. Thus, the change in computational time for different geometries will correspond with the estimated computational complexity presented in Table 1.

was found that varying the flow conditions such as the magnitude, angle of attack, and air density of the freestream velocity had no noticeable effect on the stability of computations. While varying the FMM parameters was implemented on a planar, rectangular lifting surface with a varying large aspect ratio ranging from 4 to 16 and varying angles of attack ranging from 5 to 20 deg. The results reported in this section correspond to a planar, rectangular lifting surface that has an aspect ratio of four and is subjected to a freestream velocity with a magnitude of 125 m/s, an angle of attack of 5 deg, and an air density of 1.255 kg/m³. The wing has a chord length of 1 m with nine panels in the chordwise direction and a wing span of 4 m with 36 panels in the spanwise direction. It is growing in time. Because, at each time step, the vorticity is emitted from some edges of the body, the authors can estimate that, for the 4th time step, it is determined, one should try to reduce \( N_q \) as much as possible to stay within the required FMM accuracy. Finally, clustering parameter \( s (or l_{max}) \) should be adjusted to obtain the maximum speed. Note that variation of this parameter has a small effect on the accuracy, but there exists a strong minimum of the computational time at some intermediate values of \( s \). Theoretically, this happens when the times for computations of the sparse and dense products are equal.

D. Tuning Configuration of the FMM

It can be useful to estimate the complexity of the steps of the UVLM algorithm and its overall computational complexity with and without the FMM. Consider the case of a relatively small body mesh of size \( N_q \), which is fixed, and the mesh representing the vortex sheets of size \( N_b \), which is growing in time. Because, at each time step, the vorticity is emitted from some edges of the body, the authors can estimate that, for the 4th time step, it is growing in time. Because, at each time step, the vorticity is emitted from some edges of the body, the authors can estimate that, for the 4th time step, \( N = O(N_{b}^{1/2}) \). So, the total complexity to compute \( K \) steps is a sum of the complexities for \( k = 1, 2, \ldots, K \). In Table 1, the authors show the estimates. From this table, it can be seen that the use of the FMM changes the complexity of the entire algorithm, for which the complexity at large \( K \) grows proportionally to \( O(K^2) \) as opposed to the \( O(K^3) \) for the conventional UVLM.

E. Tuning Configuration of the FMM

It can be useful to estimate the complexity of the steps of the UVLM algorithm and its overall computational complexity with and without the FMM. Consider the case of the relatively small body mesh of size \( N_q \), which is fixed, and the mesh representing the vortex sheets of size \( N_b \), which is growing in time. Because, at each time step, the vorticity is emitted from some edges of the body, the authors can estimate that, for the 4th time step, \( N = O(N_{b}^{1/2}) \). So, the total complexity to compute \( K \) steps is a sum of the complexities for \( k = 1, 2, \ldots, K \). In Table 1, the authors show the estimates. From this table, it can be seen that the use of the FMM changes the complexity of the entire algorithm, for which the complexity at large \( K \) grows proportionally to \( O(K^2) \) as opposed to the \( O(K^3) \) for the conventional UVLM.

IV. Numerical Study

In this section, the results attained from the tuning procedure of FMM configurations described in Sec. III.D are shown. The tuning procedure of the FMM parameters was implemented on a planar, rectangular lifting surface with a varying large aspect ratio ranging from 4 to 16 and varying angles of attack ranging from 5 to 20 deg. The results reported in this section correspond to a planar, rectangular lifting surface that has an aspect ratio of four and is subjected to a freestream velocity with a magnitude of 125 m/s, an angle of attack of 5 deg, and an air density of 1.255 kg/m³. The wing has a chord length of 1 m with nine panels in the chordwise direction and a wing span of 4 m with 36 panels in the spanwise direction. It was found that varying the flow conditions such as the magnitude, angle of attack, and air density of the freestream velocity had no noticeable effect on the computational time and accuracy of the simulations. Increasing (decreasing) the aspect ratio of the rectangular plate is accompanied by a corresponding increase (decrease) in the number of vortex segments and field points required for calculations in the various UVLM processes. Thus, the change in computational time for different geometries will correspond with the estimated computational complexity presented in Table 1. Therefore, the FMM parameters found in this study are general and can be applied to cases with various flow conditions and geometries. The wall-clock times reported in the following were measured on an Intel® Xeon® CPU E3-1245 v5 (3.50 GHz) eight-core PC with 16 GB of RAM.

A. Truncation Number

The first step in fine tuning the FMM involves changing the truncation number \( p_F \) to obtain a required level of accuracy of the FMM and stability of computations. While varying \( p_F \), the order of quadrature \( N_q \) should be consistent with \( p_F \) via Eq. (15) to eliminate quadrature errors. Also, the clustering parameter \( s \) (or maximum number of levels in octree \( l_{max} \)) should be kept constant. For this study, the clustering parameter was set to \( s = 200 \) and the order of quadrature was set by Eq. (15), with a truncation number of \( p_F = 2, 4, 12, 24 \).

From Fig. 6, it can be seen that, as \( p_F \) increases, the magnitude of the \( L_1 \)-norm error of the evaluated wake velocities decreases. Because the far-field interactions are approximated by polynomials of the \( p_F \)th degree in the neighborhood of the evaluation point, the higher \( p_F \) is, the more accurate the approximations of the far-field interactions will be. At \( p_F = 12 \) and \( p_F = 24 \), the \( L_1 \)-norm error becomes relatively low at orders of \( \sim 10^{-5} \) and \( \sim 10^{-8} \), respectively. In deciding which \( p_F \) to use, the authors take into account the wall-clock time for the different \( p_F \) values shown in Fig. 7. The wall-clock time corresponding to \( p_F = 24 \) is significantly higher than that corresponding to \( p_F = 12 \) and, given that the \( L_1 \)-norm error of the evaluated wake velocities for \( p_F = 12 \) is at an acceptably low level, \( p_F = 12 \) is chosen as the best truncation number for this study.

B. Order of Quadrature

With \( p_F = 12 \) giving an \( L_1 \)-norm error of order \( \sim 10^{-5} \), the order of quadrature \( N_q \) can be reduced as much as possible to stay within the required FMM accuracy. In Fig. 8, the relative \( L_1 \)-norm error of the evaluation of wake velocities is displayed when the truncation number is set to \( p_F = 12 \), the clustering parameter is set to \( s = 200 \), and the order of quadrature varies for \( N_q = 1, 2, 3, 4 \). Because the results for \( N_q > 4 \) show no noticeable difference in accuracy with respect to those for \( N_q = 4 \), the results for \( N_q > 4 \) are not reported in this paper. Note that, other than \( N_q = 1 \), the accuracies attained for \( N_q = 2, 3, 4 \) match up well with each other. Given that the wall-clock time, shown in Fig. 9, corresponding to the different values of \( N_q \) follows the same trend with approximately the same values, it is best to use \( N_q = 2 \) because the accuracy of the evaluated wake velocities does not significantly increase as \( N_q \) increases.
Finally, the clustering parameter \( s \) (or \( l_{\text{max}} \)) is selected to obtain the maximum speed. In Table 2, the authors show the wall-clock time, accuracy, and maximum number of levels in the octree \( l_{\text{max}} \) for fixed \( p_F = 0.0136 \) and \( N_q = 2 \) and clustering parameters \( s = 100, 200, 300, 400, \) and 500, where the simulator was executed for \( K = 200 \) time steps for each case. From this table, it is noted that the variation of \( s \) has a minor effect on the
accuracy of the computed wake velocities. Also note that, as the clustering parameter increases, the maximum number of levels in the octree attained decreases. Most notably, there exists a strong minimum of the computational time at \( s = 300 \). Hence, the configurations that produce the most accurate results while also providing the largest acceleration in computational time correspond to the truncation number \( p_F = 12 \), the order of quadrature \( N_q = 2 \), and the clustering parameter \( s = 300 \).

### D. Computational Cost Reduction

In Fig. 10, the results of the speed performance of the UVLM with (for \( p_F = 12, N_q = 2, \) and \( s = 300 \)) and without the FMM for the individual time steps are depicted. In this figure, the computational time for the evaluation of the wake velocity fields of the UVLM is plotted against the number of time steps. On a logarithmic scale, the slope of the dashed line is two, which indicates the computational complexity of the standard UVLM evaluation of wake velocities for large \( K \) is of \( O(K^2) \), where \( K \) is the total number of time steps. The slope of the computational complexity of the FMM accelerated UVLM is indicated by the dotted line, which has a slope of one (i.e., grows proportionally to \( O(K) \)). Thus, with the FMM, the computational complexity of the evaluation of wake velocities reduced from \( O(K^2) \) to \( O(K) \).

Notice that the FMM accelerated UVLM has better performance than the standard UVLM after about \( k = 8 \) time steps (approximately \( N = 1500 \) sources). This improvement in speed is achieved with no noticeable loss in accuracy. The relative \( L_2 \)-norm error of the computed wake velocities of the FMM accelerated UVLM is of order \( \sim 10^{-5} \).

---

### Table 2  Wall-clock time and accuracy of FMM for different clustering parameter

<table>
<thead>
<tr>
<th>( s )</th>
<th>Wall-clock time, s</th>
<th>Maximum levels in octree</th>
<th>Average ( L_2 )-norm error</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>45.34</td>
<td>7</td>
<td>( 2.63 \times 10^{-5} )</td>
</tr>
<tr>
<td>200</td>
<td>28.06</td>
<td>6</td>
<td>( 1.86 \times 10^{-5} )</td>
</tr>
<tr>
<td>300</td>
<td>27.92</td>
<td>6</td>
<td>( 1.55 \times 10^{-5} )</td>
</tr>
<tr>
<td>400</td>
<td>29.58</td>
<td>5</td>
<td>( 1.10 \times 10^{-5} )</td>
</tr>
<tr>
<td>500</td>
<td>32.20</td>
<td>5</td>
<td>( 1.06 \times 10^{-5} )</td>
</tr>
</tbody>
</table>

Fig. 9  Wall-clock time for evaluated wake velocities for \( N_q = 1 \) (circle), \( N_q = 2 \) (square), \( N_q = 3 \) (plus sign), and \( N_q = 4 \) (triangle) with \( p_F = 12 \) and \( s = 200 \).

Fig. 10  Wall-clock time for evaluation of wake velocity in UVLM with (for \( p_F = 12, N_q = 2, \) and \( s = 300 \)) and without the FMM for the individual time steps: quadratic dependence for individual time steps of UVLM (dashed line), and linear dependence of FMM accelerated UVLM (dotted line).
Fig. 11 Wall-clock time for evaluation of wake velocity in UVLM with (for $p_f = 12, N_q = 2$, and $s = 300$) and without FMM for the cumulative time steps: cubic dependence for cumulative time steps of UVLM (dashed line), and quadratic dependence of FMM accelerated UVLM (dotted line).

In Fig. 11, the results of the speed performance of the UVLM with (for $p_f = 12, N_q = 2$, and $s = 300$) and without the FMM for the cumulative time steps are depicted. For significantly large $K$, the computational complexity of the standard UVLM evaluation of wake velocities is of $O(K^3)$, whereas the computational complexity of the FMM accelerated UVLM evaluation of wake velocities is of $O(K^2)$.

V. Conclusions

In this paper, the fast multipole method has been implemented for accelerating the unsteady vortex-lattice-method-based computational model. This is the first study wherein the fast multipole method (FMM) has been implemented in an unsteady vortex-lattice method (UVLM)-based nonlinear, unsteady aerodynamic simulator. With the FMM accelerated UVLM, the FMM parameters were tuned to obtain the necessary accuracy of evaluated wake velocities and computational cost reduction. The use of the aerodynamic model has been illustrated through the study of a planar, rectangular lifting surface with a high aspect ratio. Through this benchmark application, the authors have shown that the FMM can be successfully used to reduce the computational expense of the velocity field computations from $O(K^3)$ to $O(K)$. Calculations for the velocity fields that would normally take about 15 min for a non-FMM accelerated UVLM-based computation only take about 10 s with the FMM accelerated UVLM version. This improvement in speed is achieved with no noticeable loss in accuracy because the relative $L_2$-norm error of the computed velocities of the FMM accelerated UVLM is of order $10^{-5}$. This significant acceleration in computations allows for simulation domain sizes that were previously computationally intractable. Thus, modeling more complex systems, such as the joined-wing SensorCraft, to attain aeroelastic responses becomes practically viable. Faster simulations will allow for more simulation data to be obtained, which will help in achieving a near-real-time decision support system through the dynamic data-driven application systems paradigm. Only the velocities evaluation in the UVLM, which is the most computationally expensive portion of the aerodynamic model, is accelerated with the FMM in this paper. In the future, acceleration of other aspects of the computation in determining the aerodynamic loads will be explored. The benefits of this acceleration on aeroelastic computations will be examined, and the advantages of combining the FMM with graphics processing unit computing for aeroelastic computations will be studied.

Acknowledgments

Partial support received from the U.S. Air Force Office of Scientific Research Dynamic Data-Driven Application Systems Program through grant no. FA95501510134, the National Science Foundation through grant no. CMMI-1250187, and Fantalgo, LLC through use of the licensed fast multipole module in this study are gratefully acknowledged.

References


H.-L. Choi
Associate Editor