On Using a Fast Multipole Method-based Poisson Solver in an Approximate Projection Method

Sarah A. Williams*

Ann S. Almgren†

E. Gerry Puckett*

Abstract

Approximate projection methods are useful computational tools for solving the equations of time-dependent incompressible flow. In this report we will present a new discretization of the approximate projection in an approximate projection method. The discretizations of divergence and gradient will be identical to those in existing approximate projection methodology using cell-centered values of pressure; however, we will replace inversion of the five-point cell-centered discretization of the Laplacian operator by a Fast Multipole Method-based Poisson Solver (FMM-PS).

We will show that the FMM-PS solver can be an accurate and robust component of an approximation projection method for constant density, inviscid, incompressible flow problems. Computational examples exhibiting second-order accuracy for smooth problems will be shown. The FMM-PS solver will be found to be more robust than inversion of the standard five-point cell-centered discretization of the Laplacian for certain time-dependent problems that challenge the robustness of the approximate projection methodology.

*Department of Mathematics, University of California, Davis

†Lawrence Berkeley National Laboratory

1 Introduction

Projection methods have been used for decades to solve time-dependent incompressible flow problems. Introduced as a first-order accurate method for solving constant density incompressible flows [11], projection methods have since been extended to second-order accuracy [6], variable density flows [8; 22], inhomogeneous constraints [12; 21], approximate forms of the projection operator [5; 18], and a variety of geometric configurations including embedded boundary representations [2; 3] and adaptive mesh refinement [1; 19; 20; 26].

In all of these instantiations, however, the core of the methodology is the same. An advection-diffusion step is used to advance the velocity in time; the solution is then projected (or approximately projected) onto the constraint space. The projection operator requires solution of a Poisson equation. In the case of an exact projection, the discrete form of the Laplacian operator is dictated by the choice of discrete divergence and gradient operators. In the case of an approximate projection, there is some latitude in choosing the approximation to the Laplacian operator; it is not uniquely defined by the divergence and gradient operators. Analytically, an exact projection is preferable; however, there are numerical difficulties associated with solution of the Poisson equation dictated by an exact projection that have led to the extensive use of approximate projection methods instead. The motivation for an approximate rather than exact projection is covered in detail in [5; 18; 19].

Typically, solution of the Poisson equation dominates the computation time of the flow problem, making a robust and efficient solver an essential component of any successful projection method. This report explores the use of a Fast Multipole Method-based Poisson Solver (FMM-PS) in a second-order accurate approximate projection method for inviscid, constant density, incompressible flow. Both the projection method and the FMM-PS solver are coded in FORTRAN 90, using the object-oriented software design developed at the Center for Computational Science and Engineering at Lawrence Berkeley National Laboratory. This software framework provides easy intercompatibility with other software elements as well as easy extension from serial to massively parallel architectures.

The following sections of this report contain summaries of the projection methodology and the FMM-PS solver, a discussion of how they have been coupled in this application, and computational examples demonstrating the second-order accuracy and robustness of the new approximate projection formulation.

2 Projection Method

We consider here a projection method for solving the Euler equations for constant density, inviscid, incompressible fluid flow in a two-dimensional doubly periodic domain with no external forcing.

The incompressible Euler equations are given by

$$\mathbf{u}_t = -(\mathbf{u}\cdot\nabla)\mathbf{u} - \nabla p \tag{1}$$

$$\nabla\cdot\mathbf{u} = 0 \tag{2}$$

where $\mathbf{u}$ is the velocity and p is the pressure. For the algorithm described here the velocity is defined at cell centers and integral time levels; the pressure is also defined at cell centers but at intermediate times.

The projection method is a fractional step method. First, (1) is discretized in time to construct a new-time provisional velocity field, $\mathbf{u}^*$, without enforcing (2). Second, a projection is applied to the new-time velocity field so that the resultant field satisfies (2). The name *projection method* is used because the intermediate solution, $\mathbf{u}^*$, is *projected* onto the space of divergence-free fields.

In the original projection method developed by Chorin [11] the projection step of the algorithm is specified by defining discrete operators D and $\mathbf{G}$, approximating divergence and gradient, respectively, which are skew-adjoint; i.e., $D = -\mathbf{G}^T$. With this definition the projection operator, $\mathbf{P} = \mathbf{I} - \mathbf{G}(D\mathbf{G})^{-1}D$ (with boundary conditions implicitly defined by the the flow problem), is a discrete orthogonal projection on the finite-dimensional space of vector fields defined on the mesh. In Chorin's formulation both pressure and velocity are specified at nodes and central differences are used for the definition of D and $\mathbf{G}$. This results in an expanded five-point stencil for the discrete Laplacian, $L = D\mathbf{G}$, that must be inverted to apply the projection. This expanded stencil produces a local decoupling of the mesh points with a 2^d-dimensional kernel for $\mathbf{G}$ where d is the dimension of the problem. Bell, Colella, and Glaz [6] use a discretization of the projection that defines the velocity on cell centers and pressure at nodes. This approach produces a more compact stencil but also generates a local decoupling of the grid. Bell, Colella and Howell [7] use a fully cell-centered analog of Chorin's algorithm. This scheme exhibits a local decoupling but in the presence of Dirichlet boundary conditions the cell-centered approximation eliminates the nonconstant elements in *ker* $\mathbf{G}$. As a result of the local grid decoupling nonstandard discretizations of $D\mathbf{G}$ must be used that require specialized iterative procedures that properly respect the stencil that is used. For the schemes in which *dim ker* $\mathbf{G} > 1$ the nonconstant elements in the kernel induce additional, artificial compatibility constraints.

Almgren *et al.* [5] first introduced the notion of an approximate projection to circumvent the numerical difficulties with exact discrete projections. Approximate projections are defined by replacing the projection operator $\mathbf{P}$ by an approximation $\widetilde{\mathbf{P}} = \mathbf{I} - \mathbf{G}(L)^{-1}D$, where L is an approximation to but not identically $D\mathbf{G}$. The operator used in [5] used pressure defined on nodes. Lai [18] introduced a cell-centered approximate projection where pressure is defined at cell centers and which uses a standard five-point centered-difference stencil for the Laplacian. The D and $\mathbf{G}$ used in this report are those from [18]; a new L that also

operates on cell-centered values will be defined by the FMM-PS solver.

In order to advance the solution in time from t^n to t^{n+1}, we first construct the time-centered advective update term, $\left[(\mathbf{u}^{\mathrm{ADV}} \cdot \nabla)\mathbf{u}\right]^{n+1/2}$ using an unsplit second-order upwind predictor-corrector scheme, where $\mathbf{u}^{\mathrm{ADV}}$ is an intermediate-time edge-based advection velocity. We define

$$\mathbf{u}^* = \mathbf{u}^n - \Delta t \left[(\mathbf{u}^{\mathrm{ADV}} \cdot \nabla)\mathbf{u}\right]^{n+1/2} - \Delta t\, \mathbf{G}p^{n-1/2} \ . \tag{3}$$

where $\mathbf{G}p^{n-1/2}$ is a lagged approximation to the pressure gradient,∇p. The construction of $\left[(\mathbf{u}^{\mathrm{ADV}} \cdot \nabla)\mathbf{u}\right]^{n+1/2}$ that we use in the numerical examples in this paper is exactly that given in Appendix A of [4]; specifically, we use conservative differencing rather than the convective differencing used in [5], for example.

In the second part of the fractional step scheme, we project a vector field $\mathbf{V}$ onto the space of divergence-free fields. For an exact projection, $\mathbf{V}$ can take many forms, all of which lead to exactly the same solution (assuming an exact solution of the resulting Poisson equation). Two natural candidates for $\mathbf{V}$ are the velocity itself or the update to velocity. One could also modify each of these by removing the pressure gradient component of the update. Specifically, one might choose any of the following (labeled as in [4]):

$$\begin{aligned}
(1) \quad \mathbf{V} &= u^{*,n+1} \\
(2) \quad \mathbf{V} &= u^{*,n+1} + \Delta t\, \mathbf{G}p^{n-1/2} \\
(3) \quad \mathbf{V} &= u^{*,n+1} - u^n \\
(4) \quad \mathbf{V} &= u^{*,n+1} - u^n + \Delta t\, \mathbf{G}p^{n-1/2},
\end{aligned}$$

Then, after solving $D\mathbf{G}\phi = D\mathbf{V}$ and setting $\mathbf{V}_d = \mathbf{V} - \mathbf{G}\phi$, the new velocity and pressure would be defined, respectively, by

$$\begin{aligned}
(1) \quad u^{n+1} &= \mathbf{V}_d, & p^{n+1/2} &= p^{n-1/2} + \frac{1}{\Delta t}\phi \\
(2) \quad u^{n+1} &= \mathbf{V}_d, & p^{n+1/2} &= \frac{1}{\Delta t}\phi \\
(3) \quad u^{n+1} &= u^n + \mathbf{V}_d, & p^{n+1/2} &= p^{n-1/2} + \frac{1}{\Delta t}\phi \\
(4) \quad u^{n+1} &= u^n + \mathbf{V}_d, & p^{n+1/2} &= \frac{1}{\Delta t}\phi
\end{aligned}$$

For approximate projections the choice of $\mathbf{V}$ has a nontrivial effect on the solution. Because the approximate projection operators are second-order accurate approximations to an exact projection, the methods that result from each choice of $\mathbf{V}$ are all second-order accurate for smooth problems, but not identical. The implications of these choices are discussed and analyzed in [4] for the nodal approximate

projection operator [5] and the cell-centered operator [18]. For the purposes of this paper we consider versions (1) and (2) as above.

The discretization of the divergence and gradient operators we will consider are described in [18; 19], and are straightforward:

$$(D\mathbf{u}^*)_{i,j} = \frac{1}{2\Delta x}(u_{i+1,j} - u_{i-1,j}) + \frac{1}{2\Delta y}(v_{i,j+1} - v_{i,j-1}) \tag{4}$$

and

$$(\mathbf{G}\phi)_{i,j} = (\frac{1}{2\Delta x}(\phi_{i+1,j} - \phi_{i-1,j}), \frac{1}{2\Delta y}(\phi_{i,j+1} - \phi_{i,j-1})) \tag{5}$$

In the case of [18; 19], L^{CC} is the standard five-point approximation to the Laplacian operator,

$$(L^{CC}\phi)_{i,j} = \frac{1}{\Delta x^2}(\phi_{i+1,j} + \phi_{i-1,j} - 2\phi_{i,j}) + \frac{1}{\Delta y^2}(\phi_{i,j+1} + \phi_{i,j-1} - 2\phi_{i,j}) \tag{6}$$

We note that, discretely, $L^{CC} \neq D\mathbf{G}$, hence the approximateness of the projection. In the case of the FMM-PS solver as described below, the projection is also approximate; $L^{FMM} \neq D\mathbf{G}$, where L^{FMM} is the operator discretely used by the FMM-PS solver.

3 Fast Multipole Method-based Poisson Solver (FMM-PS)

3.1 Background

The Fast Multipole Method, originally presented as a scheme for solving boundary value problems for the Laplace equation ([24]) or for solving N-body problems ([14; 16; 10]), has also been presented as a fast, direct solver for free space and boundary value Poisson problems. The algorithm described in this report closely follows that described in [13]. The problem under consideration in [13] is the two-dimensional Poisson problem,

$$\nabla^2\phi = f, \tag{7}$$

in free space or on a square domain, with periodic, Dirichlet, or Neumann boundary conditions; an adaptively refined mesh is also used. The solution proceeds from evaluation of the fundamental solution to the free space Poisson problem, namely

$$\phi(\mathbf{x}) = \frac{1}{2\pi}\int_{\mathbb{R}^2} \log\|\mathbf{x} - \mathbf{y}\| \cdot f(\mathbf{y})\, d\mathbf{y} \tag{8}$$

where $\mathbf{x} \in \mathbb{R}^2$ and $\|\cdot\|$ denotes the distance. In the remainder of this section, we will consider $\mathbf{x}$ instead as a point in the complex plane for convenience of notation.

Other work that has used the FMM to evaluate Equation (8) includes [25; 15]. A version of the FMM-PS has been previously implemented in MATLAB and made available for public use (see www.madmaxoptics.com). Here we develop solutions to the free space and periodic domain problems, and restrict our attention to a uniform mesh. The version discussed here is implemented in FORTRAN 90, based on the BoxLib framework developed in the Center for Computational Sciences and Engineering (CCSE) at LBNL.

3.2 Overview

We consider a hierarchical grid structure covering the domain. Level 0 of this structure is a single cell covering the entire computational domain; partitioning the domain into four square cells results in level 1, and so forth, for levels $\ell = 1, ..., L$ where $2^L \times 2^L$ is the resolution at which we wish to solve the problem.

For each cell Ω_i at level $\ell > 0$ we define the *parent* as the cell at level $\ell - 1$ that contains Ω_i; we define the *child* of Ω_i at level $\ell < L$ as any one of the four cells at level $\ell + 1$ that are contained by Ω_i. By *near neighbor* of Ω_i we mean the eight cells on the same level as Ω_i that share a boundary point with Ω_i, and by *intermediate neighbor* of Ω_i we mean the 27 cells on the same level as Ω_i which are not Ω_i or its near neighbors, but which are children of Ω_i's parent or its near neighbors.

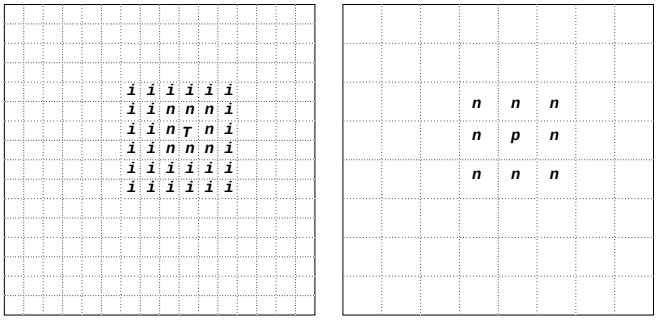


Figure 1: The left figure shows the grid at level 4. The cell T's near neighbors are marked with an n, and its intermediate neighbors are marked with an i. In the figure on the right, showing the grid at level 3, T's parent is marked with a p, and the near neighbors of p are marked with an n.

It is clear that near neighbors and intermediate neighbors of some cells lie beyond the boundary of the computational domain. We make use of ghost cells, populated with values according to the boundary conditions under consideration; see Section 3.4 below for more details.

Two types of series expansions are fundamental to the FMM. A $(p+1)$-term *multipole expansion* centered at point $\mathbf{x}_c$,

$$\psi(\mathbf{x}) = \mathrm{Re}\left\{ a_0 \log(\mathbf{x} - \mathbf{x}_c) + \sum_{k=1}^{p} \frac{a_k}{(\mathbf{x} - \mathbf{x}_c)^k} \right\} \quad , \tag{9}$$

is singular at $\mathbf{x}_c$ and accurate far from $\mathbf{x}_c$. We say that a multipole expansion is associated with a cell Ω_i

when it is centered at the center of Ω_i and describes the field due to the source function contained within Ω_i. It is not accurate in Ω_i or its near neighbors (see [14] for details), but it can be evaluated accurately beyond those cells.

A $(p+1)$-term *local expansion* centered at $\mathbf{x}_c$,

$$\phi(\mathbf{x}) = \mathrm{Re}\left\{\sum_{k=0}^{p} b_k \cdot (\mathbf{x}-\mathbf{x}_c)^k\right\} \quad , \tag{10}$$

is a truncated Taylor series with approximate coefficients. It is accurate near $\mathbf{x}_c$ and not accurate far from $\mathbf{x}_c$. A local expansion associated with a cell Ω_i is centered at the center of Ω_i and describes the field in Ω_i due to the source contained beyond Ω_i. A *preliminary* local expansion associated with Ω_i describes the field in Ω_i due to the source contained beyond Ω_i, the parent of Ω_i, and the parent's near neighbors. The *augmented* local expansion associated with Ω_i describes the field in Ω_i due to the source contained beyond Ω_i and its near neighbors.

3.3 Algorithm

The following outline is intended to serve as a roadmap for users of the routine, and to highlight the differences between previous implementations and the present one. The algorithm can be described by the following steps.

Step 1. Multipole expansion

- At the finest level, determine the multipole series associated with each cell, based on the source only within that cell.

Note: In [13], each complex multipole coefficient a_k in the expansion associated with a cell Ω_i centered at $\mathbf{x}_c$ is found via numerical evaluation of an integral over Ω_i:

$$\begin{aligned} a_0 &= \frac{1}{2\pi}\int_{\Omega_i} f(\mathbf{y})d\mathbf{y} \\ a_k &= -\frac{1}{2\pi}\int_{\Omega_i} \frac{(\mathbf{y}-\mathbf{x}_c)^k f(\mathbf{y})}{k} d\mathbf{y} \end{aligned}$$

(see Equation (13) in [13] for details). In the present implementation, since the projection method algorithm as described here defines a single value within each cell for the right hand side of the Poisson equation, the

construction of the multipole coefficients can be simplified to

$$a_0 = f_i \cdot \frac{1}{2\pi} h^2 \tag{11}$$

$$a_k = -f_i \cdot \frac{1}{2\pi} \int_{\Omega_i} \frac{(\mathbf{y} - \mathbf{x}_c)^k}{k} d\mathbf{y}, \tag{12}$$

where h^2 is the volume of a small cell. The value of the integral in Equation (12) can be pre-computed.

Step 2. Child-to-parent shift

- Shift the center of each multipole expansion from child cell to parent cell via T_{cp}, the child-to-parent shift operator. T_{cp} is defined by $\{\alpha\} = T_{cp}\{a\}$, where

$$\alpha_l = \left(\sum_{k=1}^{l} a_k (\mathbf{x}_c - \mathbf{x}_p)^{l-k} \binom{l-1}{k-1} \right) - \frac{a_0 (\mathbf{x}_c - \mathbf{x}_p)^l}{l}, \; l = 0 \ldots p. \tag{13}$$

  Here, $\mathbf{x}_c$ and $\mathbf{x}_p$ are the centers of the child and parent cells, respectively; four child contributions are summed for each parent cell.

- Repeat for each successively coarser level, up to level zero.

Notes: Our implementation follows [14].

This concludes the upward pass.

When considering the downward pass, the reader might imagine that we seek a solution to the Poisson problem at only one target point in the domain. Call the target point T, and denote by $B_{T\lambda}$ the cell that contains it on level λ. In practice, every cell on finest level L contains a target at its center, and the following operations are performed for all cells on each given level simultaneously.

Step 3. Coarse-level local expansions

- Suppose that we have a local expansion associated with B_{T0} (i.e., a local expansion associated with the entire computational domain) and also a local expansion associated with B_{T1}. We consider the derivation of these coarse-level local expansions in detail in Section 3.4 below.

Step 4. Preliminary fine-level local expansions

- Shift the center of B_{T1} to the center of B_{T2} via the parent-to-child shift operator, T_{pc}, to form the preliminary local expansion associated with B_{T2}. T_{pc} is defined by $\{\beta\} = T_{pc}\{b\}$, where

$$\beta_l = \sum_{k=l}^{n} b_k \binom{k}{l} (\mathbf{x}_c - \mathbf{x}_p)^{k-1},\ l = 0 \ldots p. \tag{14}$$

Here, $\mathbf{x}_c$ and $\mathbf{x}_p$ are again the centers of the child and parent cells, respectively; this shift is performed once for each of the four children of the parent cell.

Step 5. Augmented local expansions

- Identify the intermediate neighbors of B_{T2}. (See Section 3.4 below for details on intermediate neighbors that lie beyond the boundary of the computational domain.)
- Transform the multipole expansion associated with each intermediate neighbor of B_{T2} into a local expansion centered at B_{T2}'s center, via operator T_{ml}. T_{ml} is defined by $\{b\} = T_{ml}\{a\}$, where

$$b_0 = a_0 \log(\mathbf{x}_m - \mathbf{x}_c) + \sum_{k=1}^{p} \frac{(-1)^k a_k}{(\mathbf{x}_c - \mathbf{x}_m)^k},\ \text{and} \tag{15}$$

$$b_l = \frac{-a_0}{l \cdot (\mathbf{x}_c - \mathbf{x}_m)^l} + \frac{1}{(\mathbf{x}_c - \mathbf{x}_m)^l} \sum_{k=1}^{p} \frac{a_k}{(\mathbf{x}_c - \mathbf{x}_m)^k} \binom{l+k-1}{k-1} (-1)^k,\ l = 1 \ldots p. \tag{16}$$

Here, $\mathbf{x}_m$ is the center of the multipole expansion and $\mathbf{x}_c$ is the center of the local expansion.

Sum the coefficients of these local expansions, and the coefficients of the preliminary local expansion associated with B_{T2}, to obtain the augmented local expansion associated with B_{T2}.

- Repeat Steps 4 and 5 for each successively finer level.

Notes: In [13] an accelerated T_{ml} operator is used, as introduced in [17]. For the present implementation, we use the original version of the T_{ml} operator, as described in [14].

Step 6. Evaluation of $\phi(T)$

- Evaluate the augmented local expansion associated with B_{TL} at T.
- Add in the contributions of B_{TL}'s near neighbors, and of B_{TL} itself. (See Section 3.4 below for details on near neighbors that lie beyond the boundary of the computational domain.)

Notes: 1) For use in the projection method, the only points at which we need to evaluate ϕ at the finest level are the cell centers; this reduces to $\phi = b_0$ in each cell. On the finest level, then, only the b_0 coefficient must be calculated.

2) Each near neighbor contribution depends on the value of f in the near neighbor cells, and is calculated, using pre-computed values of definite integrals, via Lemma 3.2 of [13]. We refer the reader to that paper for a discussion of the procedure.

3.4 Boundary Conditions

For the free space problem, we assume that the support of the source function f (i.e., the right-hand side of Equation (7)) is contained within the computational domain. Therefore, for all ghost cells (required in Step 5 and Step 6 of the algorithm above), the value of f and the value of each multipole coefficient is zero. Furthermore, the multipole coefficients associated with coarse cells B_{T0} and B_{T1}, as discussed in Step 3 above, are zero.

For the periodic boundary problem to be solvable, the integral of the source function f over the computational domain must be zero. The strategy for implementing the periodic boundary condition is the method of images, as described in [13]. To determine the values of f and the values of the multipole coefficients associated with ghost cells, we imagine the plane tiled with copies of the original domain.

The multipole coefficients associated with coarse cell B_{T0} (discussed in Step 3 above) are found by evaluating an infinite lattice sum first described in [23]. The details are also discussed in [14]; we use the values of the lattice sums as given in [9]. The multipole coefficients associated with B_{T1} are then found by implementing Steps 4 and 5 above.

The implementation of periodic boundary conditions is the foundation for implementation of Dirichlet and Neumann boundary conditions. The reader who wishes to extend the current implementation to address those cases is referred to [13].

3.5 Numerical Validation

Here we document the accuracy and $O(N)$ scaling in time of the method for a smooth problem on the unit square with doubly periodic boundary conditions. The analytical expression for $f(x, y)$ is given by

$$f(x, y) = \sin(2\pi x)\sin(2\pi y), \tag{17}$$

and the exact solution to $\nabla^2\phi = f$ is given by

$$\phi_{ex}(x, y) = \frac{1}{8\pi}\sin(2\pi x)\sin(2\pi y). \tag{18}$$

For the purposes of this test, we view the source function as piecewise constant over cells with the value of f at the cell center. Because of this approximation, which is analogous to using a single value for $D\mathbf{V}$ in each cell in the next section, we expect that the convergence rate of the FMM-PS will be second order rather than the higher order rate one would expect with a higher-order representation of f.

Table 17 shows the L^1, L^2 and L^∞ norms of the differences between the calculated and exact solutions at different resolutions; the calculated solution is compared against the value of ϕ_{ex} at the cell centers. Here we use thirteen terms ($p = 0, ..., 12$) in the multipole expansion. The ratios of the norms at differing resolutions confirm second-order accuracy of the method. The final three columns show the roughly $O(N)$ time-scaling behavior of the method.

Table 1: The discrete norm of the difference between the FMM solution (p = 12) and the exact solution.

grid	L^1	ratio	L^2	ratio	L^{Inf}	ratio	sec.	sec./N	ratio
64×64	4.12e-06	-	5.07e-06	-	1.01e-05	-	7.83e-01	1.91e-04	-
128×128	1.03e-06	4.00	1.27e-06	4.00	2.54e-06	3.99	3.53	2.16e-04	1.13
256×256	2.56e-07	4.02	3.17e-07	4.01	6.34e-07	4.00	1.53e+01	2.33e-04	1.08
512×512	6.30e-08	4.07	7.81e-08	4.05	1.57e-07	4.03	7.05e+01	2.69e-04	1.15

4 Using the FMM-PS in an Approximate Projection

The incorporation of the FMM-PS solver into the existing approximation projection method is straightforward. The source, f, for the solver is constructed using D and $\mathbf{V}$ as described in Section 2; this source is passed to the FMM-PS as an array of point values which can be interpreted as cell averages of f as discussed in Section 3. The FMM-PS returns a single value of ϕ for each cell; the $\mathbf{G}$ operator is then used as in Section 2 to construct $\mathbf{V}_d = \mathbf{V} - \mathbf{G}\phi$. Again we use thirteen terms ($p = 0, ..., 12$) in the multipole expansion.

Here we revisit the some of the test problems and discussion about approximate projections from [4]. First, we confirm the second-order accuracy of the methodology with the simple time-dependent diagonally translating vortices problem as used in [4]. The initial data, $\mathbf{u} = (u, v)$ is given by

$$\begin{aligned} u(x, y) &= 1 - 2\cos(x)\sin(y) \\ v(x, y) &= 1 + 2\sin(x)\cos(y) \end{aligned}$$

on the square domain, $2\pi \times 2\pi$. The exact solution to the Euler equations with this initial data is

$$\begin{aligned} u_{ex}(x,y,t) &= 1 - 2\cos(x-t)\sin(y-t), \\ v_{ex}(x,y,t) &= 1 + 2\sin(x-t)\cos(y-t). \end{aligned}$$

In the table below we present the error in u at $t = 8$ (the same final time as shown in Figure 1 of [4]) for versions (1) and (2) of $\mathbf{V}$. We include results for L^{FMM} and for L^{CC}, solved using multigrid. All calculations were run with CFL = 0.9. We note two things from these results: first, that the approximate projection

Resolution	L^{CC} (1)	ratio	L^{CC} (2)	ratio	L^{FMM} (1)	ratio	L^{FMM} (2)	ratio
32×32	8.85e-2	-	9.07e-2	-	8.95e-2	-	9.73e-2	-
64×64	2.05e-2	4.32	2.08e-2	4.36	2.06e-2	4.34	2.15e-2	4.53
128×128	4.61e-3	4.45	4.65e-3	4.47	4.62e-3	4.46	4.69e-3	4.58

Table 2: The L_2 norm of the difference between u and u_{ex} at $t = 8$ for the translating vortices problem, and the ratio of errors at differing resolutions.

method, with either L^{CC} or L^{FMM}, is second-order accurate for this smooth problem, and second, that version (1) is slightly more accurate than version (2). This is consistent with the results in [4], where it was found that the divergence of the velocity field is a useful metric for diagnosing approximation projections, and that version (1) typically has better divergence properties than version (2). This implies that one might like to use version (1) if possible. However, as we will see in the next example, version (2) tended to be more robust for the two (cell-centered and nodal) discretizations of the approximate projection that were considered. As illustrated above and discussed in [4], the difficulty with using approximate projections is not a question of formal accuracy on smooth problems. Instead, problems tend to appear as a buildup of "noise" during longer time integrations of more complex problems. To illustrate this behavior we revisit the test problem with a random initial distribution of vorticity from [4]. A key feature of this problem is that it contains substantially more high frequency content than the previous test case. The initial data for this problem are given by specifying a random stream function with spectral characteristics defined by

$$\hat{\psi}(k) = \frac{\omega}{|k|(1 + (|k|/6)^4)}$$

where ω is a normally distributed random variable mapped into the complex domain with the appropriate symmetries so that the inverse Fourier transform of $\hat{\psi}$ is a real function, ψ. We then define $\mathbf{u}$ at $t = 0$ as the discrete curl of ψ.

We examine the evolution of the solution from time $t = 0$ to 10, over which time the initial ran-

dom vorticity essentially coalesces into two smooth patches of counter-rotating vorticity. The problem is sufficiently unstable that we do not expect the locations of the resultant smooth patches to match for all methods; however, one does expect the vorticity to coalesce. The domain is the unit square with doubly periodic boundary conditions, and all calculations are run with CFL = 0.9. In Figure 1 we show raster plots of the vorticity at $t = 0$, followed by the vorticity at time $t = 10$ using versions (1) and (2) with L^{CC} and L^{FMM}.

We see here that the use of L^{FMM} appears to be more robust than the use of L^{CC} for version (1). This opens the possibility that version (1) might be more widely usable with L^{FMM} than with L^{CC}, resulting in a more accurate yet robust method for the types of problems which tend to challenge the approximate projection methodology.

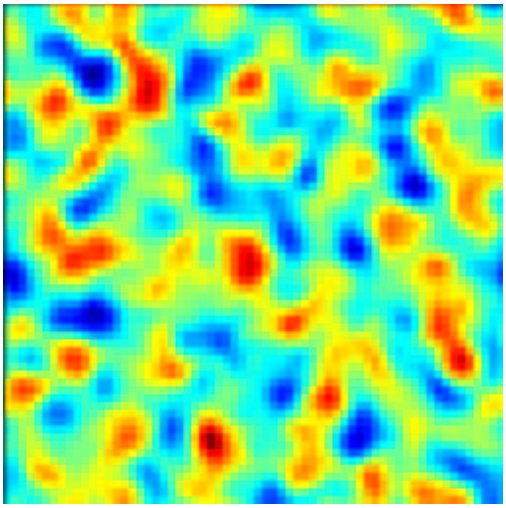
5 Concluding Remarks

We have presented a new formulation of the approximate projection in an approximate projection method. The discretizations of divergence and gradient are identical to those in existing approximate projection methodology using cell-centered values of pressure; however, inversion of the five-point cell-centered discretization of the Laplacian operator is replaced by a Fast Multipole Method-based Poisson Solver (FMM-PS).

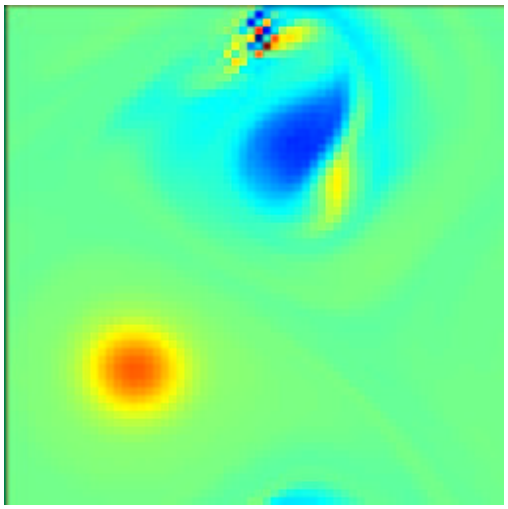
The FMM-PS solver has been shown to be an accurate and robust component of an approximation projection method for constant density, inviscid, incompressible flow problems. Timings indicate that, like other modern solvers, the solver scales linearly with the number of mesh points. Computational examples demonstrate second-order accuracy for smooth problems. The FMM-PS solver was found to be more robust than inversion of the standard five-point cell-centered discretization of the Laplacian for certain time-dependent problems which challenge the robustness of the approximate projection methodology.

6 Acknowledgements

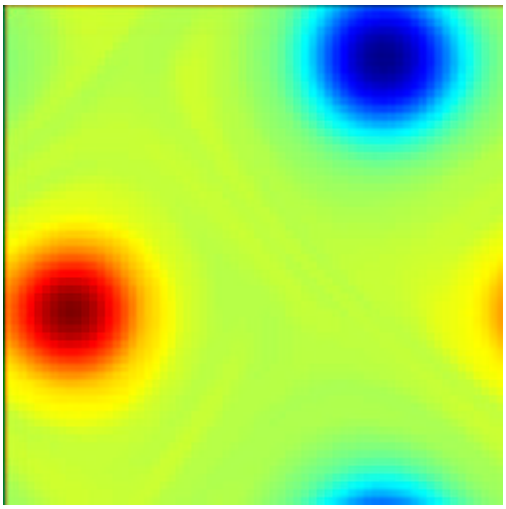
This work was supported by the Applied Mathematics Program of the DOE Office of Mathematics, Information, and Computational Sciences under the U.S. Department of Energy under contract No. DE-AC02-05CH11231.



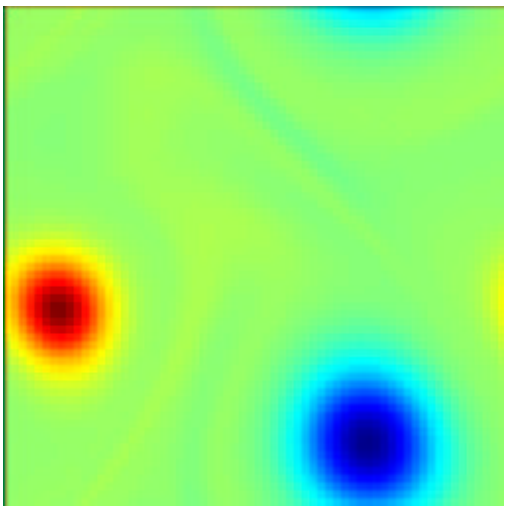
(a) $t = 0$



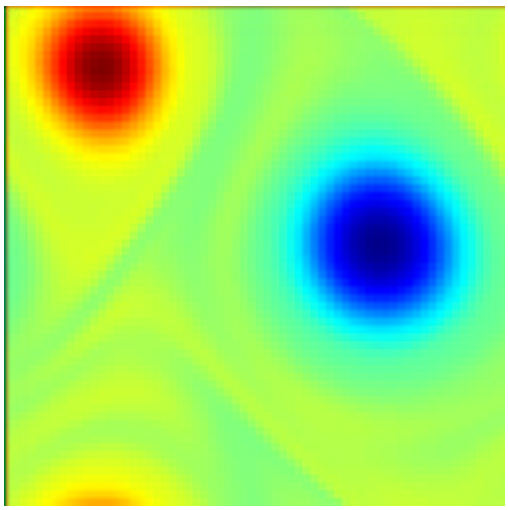
(b) L^{CC}, version (1)



(c) L^{CC}, version (2)



(d) L^{FMM}, version (1)



(e) L^{FMM}, version (2)

Figure 2: Vorticity at (a) $t = 0$, and (b)-(e) $t = 10$, for the random initial vorticity calculation at resolution 64^2.

References

[1] A. S. Almgren, J. B. Bell, P. Colella, L. H. Howell, and M. L. Welcome, *A conservative adaptive projection method for the variable density incompressible Navier-Stokes equations*, Journal of Computational Physics **142** (1998), 1–46.

[2] A. S. Almgren, J. B. Bell, P. Colella, and T. Marthaler, *A cell-centered cartesian grid projection method for the incompressible euler equations in complex geometries*, Proceedings of the 12th AIAA Computational Fluid Dynamics Conference, June 19-22 1995, San Diego, CA.

[3] ———, *A cartesian grid projection method for the incompressible euler equations in complex geometries*, SIAM J. Sci. Comput. **18** (1997), no. 5, 1289–.

[4] A. S. Almgren, J. B. Bell, and W. Y. Crutchfield, *Approximate projection methods: Part I. Inviscid analysis*, SIAM J. Sci. Comput. **22** (2000), no. 4, 1139–59.

[5] A. S. Almgren, J. B. Bell, and W. G. Szymczak, *A numerical method for the incompressible Navier-Stokes equations based on an approximate projection*, SIAM J. Sci. Comput. **17** (1996), no. 2, 358–369.

[6] J. B. Bell, P. Colella, and H. M. Glaz, *A second order projection method for the incompressible Navier-Stokes equations*, Journal of Computational Physics **85** (1989), no. 2, 257–283.

[7] J. B. Bell, P. Colella, and L. H. Howell, *An efficient second-order projection method for viscous incompressible flow*, Proceedings of the Tenth AIAA Computational Fluid Dynamics Conference, AIAA, June 1991, pp. 360–367.

[8] J. B. Bell and D. L. Marcus, *A second-order projection method for variable-density flows*, Journal of Computational Physics **101** (1992), no. 2, 334–348.

[9] C. L. Berman and L. Greengard, *A renormalization method for the evaluation of lattice sums*, J. Math. Phys. **35** (1994), 6036–6048.

[10] J. Carrier, L. Greengard, and V. Rokhlin, *A fast adaptive multipole algorithm for particle simulations*, SIAM J. Sci. Stat. Comput. **9** (1988), no. 4, 669–686.

[11] A.J. Chorin, *A numerical method for solving incompressible viscous flow problems*, J. Comput. Phys. **2** (1967), no. 1, 12–26.

[12] M. S. Day and J. B. Bell, *Numerical simulation of laminar reacting flows with complex chemistry*, Combust. Theory Modelling **4** (2000), no. 4, 535–556.

[13] Frank Ethridge and Leslie Greengard, *A new Fast-Multipole accelerated Poisson solver in two dimensions*, SIAM J. Sci. Comput. **23** (2001), no. 3, 741–760.

[14] L. Greengard and V. Rokhlin, *A fast algorithm for particle simulations*, J. Comput. Phys. **73** (1987), 325.

[15] Leslie Greengard and June-Yub Lee, *A direct adaptive Poisson solver of arbitrary order accuracy*, J. Comput. Phys. **125** (1996), 415–424.

[16] Leslie F. Greengard, *The rapid evaluation of potential fields in particle systems*, ACM Distinguished Dissertation, MIT Press, 1987, Cambridge, MA.

[17] T. Hrycak and V. Rokhlin, *An improved fast multipole algorithm for potential fields*, SIAM J. Sci. Comput. **19** (1998), 1804–1826.

[18] M. F. Lai, *A projection method for reacting flow in the zero Mach number limit*, Ph.D. thesis, University of California, Berkeley, September 1993.

[19] D Martin and P Colella, *A cell-centered adaptive projection method for the incompressible Euler equations*, J. Comput. Phys (2000), 271–312.

[20] M.L. Minion, *A projection method for locally refined grids*, J. Comput. Phys. **127** (1996), 158–178.

[21] R. B. Pember, L. H. Howell, J. B. Bell, P. Colella, W. Y. Crutchfield, W. A. Fiveland, and J. P. Jessee, *An adaptive projection method for unsteady low-Mach number combustion*, Comb. Sci. Tech. **140** (1998), 123–168.

[22] Elbridge G. Puckett, Ann S. Almgren, John B. Bell, Daniel L. Marcus, and William G. Rider, *A higher-order projection method for tracking fluid interfaces in variable density incompressible flows*, Journal of Computational Physics **130** (1997), 269–282.

[23] Lord Rayleigh, *On the influence of obstacles arranged in a rectangular order upon the properties of the medium*, Philos. Mag. **34** (1892), 481–502.

[24] V. Rokhlin, *Rapid solution of integral equations of classical potential theory*, Journal of Computational Physics **60** (1985), no. 2, 187–207.

[25] G. Russo and J.A. Strain, *Fast triangulated vortex methods for the 2-D Euler equations*, J. Comput. Phys. **111** (1994), 291–323.

[26] M. M. Sussman, A. S. Almgren, J. B. Bell, P. Colella, L. Howell, and M. Welcome, *An adaptive level set approach for incompressible two-phase flows*, Journal of Computational Physics **148** (1999), 81–124.