The Numerical Solution of the *N*-Body Problem

Leslie Greengard

Citation: Computers in Physics **4**, 142 (1990); doi: 10.1063/1.4822898
View online: https://doi.org/10.1063/1.4822898
View Table of Contents: https://aip.scitation.org/toc/cip/4/2
Published by the American Institute of Physics

The Numerical Solution of the *N*-Body Problem

Leslie Greengard

The N-body problem of gravitation or electrostatics has interested astronomers, physicists and mathematicians for the last several hundred years. It has, in fact, motivated much of the theoretical work done during that time, from cosmology to solid state physics to differential equations and potential theory. Unfortunately, the equations of motion of systems with more than two particles do not allow for an analytical solution. Until recently, therefore, theoretical studies have been limited to determining the coarse qualitative behavior of the system. With the introduction of high-speed computers, however, we have a tool which allows us to look at the trajectories of many particles in detail. The use of computer simulation to study particle systems is growing in a wide variety of fields.

The equations of motion of a system of N particles are described by a set of N ordinary differential equations, each of which is simply Newton's second law of motion applied to an individual particle. Given initial positions and velocities, a computer simulation follows all the trajectories by numerical integration. The task at every time step is to move each particle according to its velocity, and then to update its velocity according to the force exerted by the other particles. It is clear that the amount of work required for moving all particles is proportional to N, but that the amount of work required for computing all forces is proportional to N^2, since it involves determining all pairwise interactions. This quadratic growth in cost has been a major limitation on the size of the systems which can be handled, even by the fastest supercomputers. Many problems of interest require millions of particles and have been well beyond reach.

In this article, we will refer to the determination of all pairwise forces (or potentials) for a fixed configuration as the *numerical N-body problem*. In the last twenty years, significant progress has been made in reducing the cost of this calculation. The first breakthrough, allowing simulations with orders of magnitude more bodies, came with the advent of particle/mesh (PM) methods. These have been described in great detail by Hockney and Eastwood. They are extremely efficient when the particle distribution is more or less uniform and the required precision is relatively low. There is a severe degradation of performance when the sources are clustered and when the required precision is high[1].

In the last few years, a group of algorithms has been developed in the astrophysics community which have come to be known as "tree codes" or "hierarchical codes." They are due to Appel, Barnes and Hut, and others. They are designed to work well in a variety of settings, including ones where there is a high degree of clustering. The basic idea is to replace groups of distant particles by their centers of mass, and to compute the interactions between groups via this approximation. Although some accuracy is generally sacrificed, the amount of time required by these methods grows as $N \log N$ rather than N^2. Simultaneous with the development of these tree codes, while working in fluid dynamics, Van Dommelen and Rundensteiner proposed a scheme for two-dimensional calculations which is also $N \log N$, but highly accurate. Although the physical intuition underlying their method is the same, the increased accuracy is obtained by the application of

Leslie Greengard is Assistant Professor at Courant Institute, New York University. His research interests are in fast algorithms for particle simulation and integral equations.

asymptotic analysis. A closely related scheme, the Fast Multipole Method (FMM) has been developed by Greengard and Rokhlin. It is also highly accurate, involves a more complicated asymptotic analysis, works in both two and three dimensions, and requires an amount of work proportional to N. Before describing these algorithms, we illustrate the ubiquity of numerical N-body calculations by giving several examples of actual applications (Fig.1).

Astrophysics

Astrophysics is a field in which computer simulation now plays a major role (see Fig.2). There are vast numbers of problems concerning the dynamical behavior of large systems of stars which can be approached in no other way. In fact, much of the early work on the numerical solution of the N-body problem grew out of the need for more efficient methods to handle both the large number of masses and the long time intervals involved in such calculations. Although it is not our intention to provide a comprehensive discussion of the N-body techniques employed in astronomical problems, we indicate several of the main developments in Fig.3.

Plasma Physics

Another rich source of N-body problems is plasma physics. The bodies in this case are ions and electrons, and one integrates the equations of motion of a large number of such particles in their self-consistent electric and magnetic fields. To date, the vast majority of such simulations have employed particle/mesh methods and have greatly enhanced our understanding of many of the complex phenomena which characterize such systems. However, there are several interesting problems which have been essentially unapproachable with these methods due to numerical instabilities arising from the grid. Dynamical simulations of "cold plasmas" and ion beams, for example, cannot be followed over the time intervals of interest. The fast N-body methods discussed in this article are gridless, and should provide a means for overcoming these obstacles.

Molecular Dynamics

Molecular dynamics is a technique for studying the properties of matter by computer simulation. The idea is to choose a collection of particles representing individual atoms or molecules which constitute the material under consideration. The resulting simulation is then used much like experimental data to compute properties of interest (correlation functions, structure of solvent around solute, dielectric constant, etc.)

In early work, only non-polar fluids were considered, with either "hard-sphere" particle-particle interactions, or interactions governed by a Lennard-Jones potential

$$\Phi(r_{ij}) \propto \left(\frac{\sigma}{r_{ij}}\right)^{12} - \left(\frac{\sigma}{r_{ij}}\right)^{6}. \qquad (1)$$

At long range, this interaction has an attraction proportional to r^{-6}, while at close range, there is an r^{-12} repulsion. Because of the rapid decay, interactions need only be accounted for up to a fixed truncation radius. As a result, the amount of computation per time step is proportional to the number of particles N.

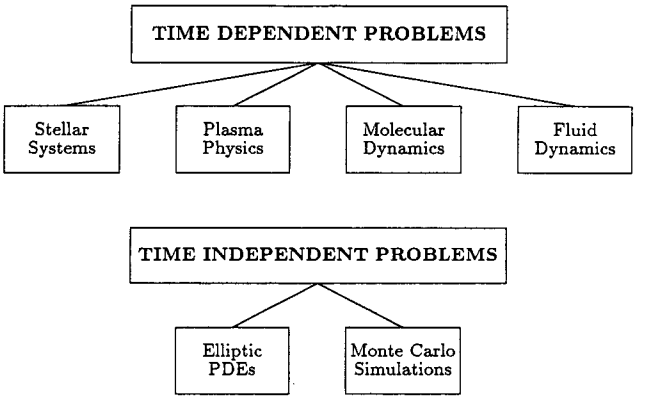


Fig.1: APPLICATIONS OF THE NUMERICAL N-BODY PROBLEM. In time-dependent problems, the interactions are used to compute a force field (stellar systems, plasmas, molecular dynamics) or a velocity field (fluid dynamics). In static problems, the N-body calculation is used either to measure the potential energy of a configuration (Monte Carlo simulations) or as part of a more complicated iterative procedure to compute the solution of a boundary value problem.

In polar fluids, the situation is quite different. A Coulombic term is added to the potential function Φ in (1), which introduces an electrostatic field. In a fully self-consistent simulation, all pairwise interactions should be accounted for. The usual approach taken in simulations of polar fluids, however, has been to truncate the potential at some fixed cut-off distance. Although very little error seems to be introduced in the pairwise correlation functions, dielectric properties are poorly simulated. The answer to the question of why these dielectric properties are incorrectly computed is currently unknown. The difficulty may be in truncation, it may be that the number of particles has not reached the asymptotic range, and it may be that the semi-classical molecular dynamics model is insufficient. Fast N-body methods will allow for fully self-consistent simulations without truncation, and should help to distinguish between these cases.

Elliptic Partial Differential Equations

It is well-known that boundary value problems for the Laplace equation can be reduced to integral equations of the second kind by means of classical potential theory[2]. For example, to solve the Dirichlet problem

$$\nabla^2\Phi(p) = 0 \text{ in } \Omega$$
$$\Phi(p) = f(p) \text{ on } \partial\Omega,$$

we try to determine a function $\sigma(t)$ such that

$$\Phi(p) = \int_{\partial\Omega} G(p,t) \cdot \sigma(t)\, dt,$$

where $G(p,t)$ is the potential at p due to a normally oriented dipole of unit strength at t. To satisfy the boundary condition, the unknown dipole density $\sigma(t)$ must satisfy

$$\pi \cdot \sigma(p) - \int_{\partial\Omega} G(p,t) \cdot \sigma(t)\, dt = f(p). \qquad (2)$$

In order to solve (2), we discretize the boundary $\partial\Omega$ into n nodes, converting the integral equation into a dense linear

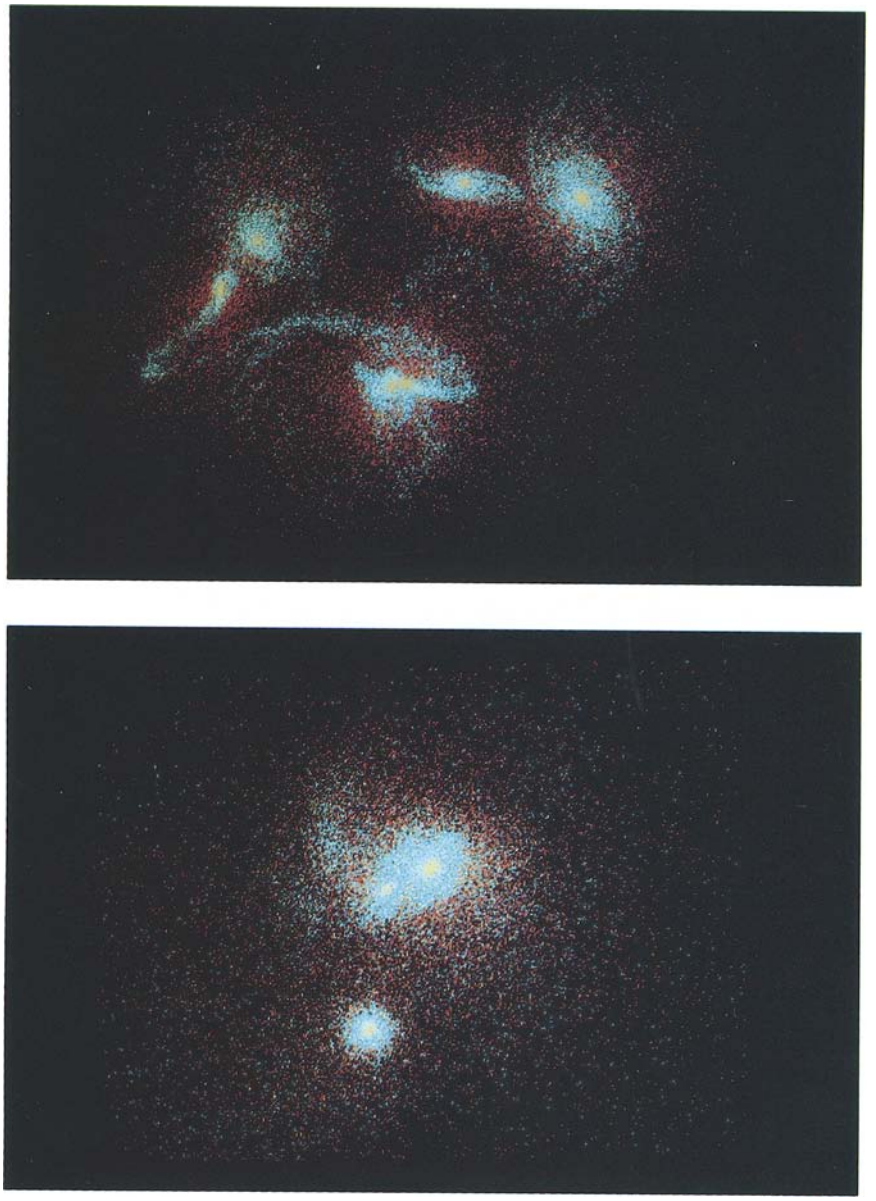


Fig.2: INTERACTING GROUP OF GALAXIES. This sequence of images shows the merger of a group of galaxies. The central galactic bulges are represented in yellow, while the surrounding disks of stars are blue. The dark halo of matter present in the galaxies is marked in red. It cannot be seen, but has profound gravitational effects. All components are modeled as fully three-dimensional N-body systems with thousands of particles each. The Barnes-Hut tree code, with its adaptive structure, made the simulation possible without imposing any artificial restrictions on the physical extent or spatial scales present in the system. The simulation is by Joshua Barnes at the Institute for Advanced Study. The results obtained support the view that compact groups are short-lived systems which eventually merge to form elliptical galaxies.

system of order n, which can in turn be solved iteratively. Each iteration requires the application of the matrix to a vector, and applying a dense matrix to a vector requires order $O(n^2)$ operations. In this case, however, the matrix-vector product corresponds to evaluation of the field due to n dipoles located on the boundary at each of the source locations themselves—a numerical N-body problem. It is possible, by using the FMM, to apply the matrix to a vector in order $O(n)$ operations and to solve the linear system in an amount of time proportional to n. Once the integral equation for $\sigma(t)$ has been solved, the solution Φ to the original differential equation can be evaluated at m interior points in order $O(m+n)$ operations. This approach to the solution of boundary value problems was originally developed by Rokhlin. The FMM is, in fact, an extension to more general distributions of particles and to higher space dimensions of the algorithm he employed for the calculation of N-body interactions of charges located on a curve in the plane.

Fluid Dynamics

One area where elliptic partial differential equations play an important role is fluid dynamics. If the flow is incompressible and irrotational, the velocity field can be obtained as the gradient of a potential function which satisfies Laplace's equation. If the flow is incompressible and viscous, the velocity field can be obtained as the curl of a stream function Ψ which satisfies Poisson's equation

$$\nabla^2\Psi = -\xi, \tag{3}$$

where ξ is the vorticity.

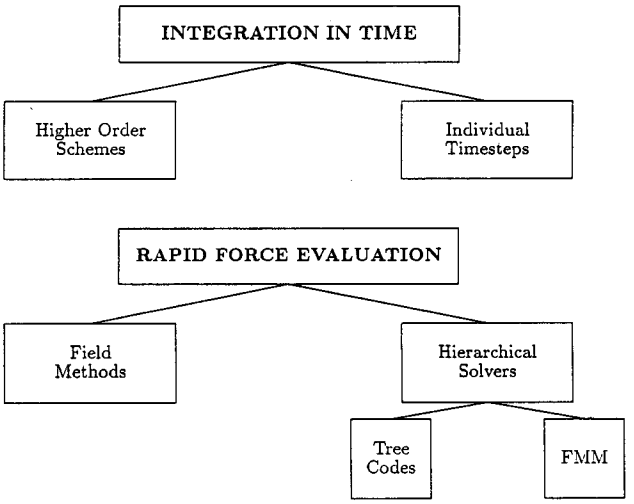


Fig.3: PROGRESS IN THE NUMERICAL SOLUTION OF THE GRAVITATIONAL *N*-BODY PROBLEM. Early work by Aarseth and others concentrated on refining the time-marching algorithms. By viewing the net force as having a near and far component, for example, it became possible to efficiently move each particle according to its own time step. Later work concentrated more on the numerical *N*-body problem. Field methods were introduced around 1970, while the hierarchical solvers came into being only in the 1980's.

An interesting application of numerical *N*-body calculations is Chorin's vortex method, in which particles ("vortex blobs") are used to discretize the vorticity field. (See Fig.4 a,b). The motion of these particles is determined by the Navier-Stokes equations. The dominant cost per time step is the evaluation of the interactions between N vortex blobs to determine the velocity field. These interactions are governed by Poisson's equation and, hence, are Coulombic.

The Numerical *N*-body Problem

We turn now to a description of the hierarchical algorithms. Although all results carry over into three-dimensional systems, the necessary analysis is substantially simpler in two dimensions. The model we consider, therefore, is the numerical N-body problem in the complex plane, viewed as electrostatics. That is, given the positions z_i and strengths q_i of N charged particles, we wish to determine the net potential and electric field at each particle position using Coulomb's law. These are given by the expressions

$$\phi(z_i) = \sum_{j \neq i} q_j \cdot \log(z_i - z_j) \tag{4}$$

and

$$E(z_i) = \phi'(z_i) = \sum_{j \neq i} \frac{q_j}{(z_i - z_j)} \tag{5}$$

respectively[3]. The fast algorithms rely upon replacing certain portions of the direct summations above by series expansions.

A First Order Approximation

Consider the situation depicted in Fig.5 and let z_i be the position of one of the charges in disk A. Since $|z| > |z_i|$, the following expansions are valid:

$$q_i \cdot \log(z - z_i) = q_i \cdot \log z - \frac{q_i \cdot z_i}{z} + O\left(\frac{z_i}{z}\right)^2 \tag{6}$$

$$\frac{q_i}{(z - z_i)} = \frac{q_i}{z} + \frac{q_i \cdot z_i}{z^2} + O\left(\frac{z_i}{z}\right)^2 \tag{7}$$

The terms in the second expansion are clearly the derivatives of the terms retained in the first, as we would expect from (5). The error in using these approximations is second order in the expansion parameter $\frac{z_i}{z}$, which is bounded above by $\frac{R}{R+D}$.

Suppose now that we want to evaluate the field at z due to all the charges contained in disk A of Fig.5. From (6) and (7), we have

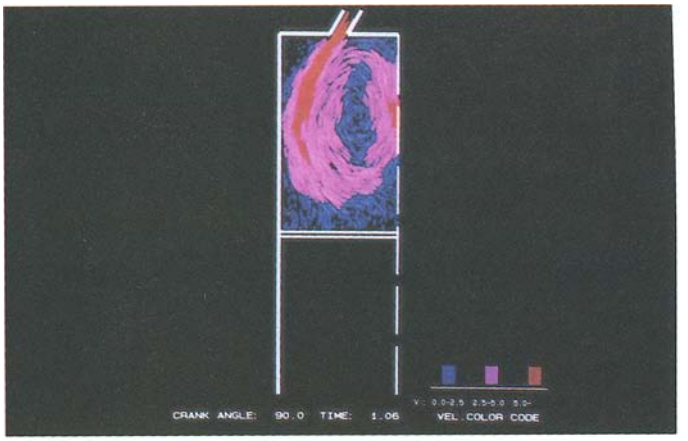
$$\phi(z) = \Sigma q_i \log(z - z_i) = (\Sigma q_i) \cdot \log z - \frac{(\Sigma q_i \cdot z_i)}{z} + O\left(\frac{R}{z}\right)^2 \tag{8}$$

$$E(z) = \Sigma \frac{q_i}{(z - z_i)} = \frac{(\Sigma q_i)}{z} + \frac{(\Sigma q_i \cdot z_i)}{z^2} + O\left(\frac{R}{z}\right)^2 \tag{9}$$

The summations are over all n sources in A. The first coefficient $(\Sigma q_i)$is the net charge, while the second $(\Sigma q_i \cdot z_i)$ is the dipole moment. If we simply wanted to determine the potential ϕ and the electric field E at a single point, there is little benefit to be obtained from forming these expansion coefficients. If we wanted to sample ϕ and E at n distant points, however, then a direct calculation would require n^2 computations. On the other hand, we could first compute the net charge and dipole moment, and then evaluate the expressions (8) and (9) at each sampling point z. The amount of work using expansions is proportional to n, a significant savings. In numerical N-body calculations, the sources and sampling points are not so clearly separated, but by combining the use of expansions with a "divide and conquer" strategy, it is possible to develop efficient and effective algorithms. This is the approach taken by the tree codes.

The *N* log *N* Tree Code

We begin our discussion with what is essentially a two-dimensional non-adaptive version of the Barnes/Hut scheme. The notation we introduce may seem unnecessarily complex, but it will carry us over into our description of the other N-body algorithms. (The original three-dimensional tree-codes have a slightly different form. Rather than compute the first two moments of the expansion, as above, they compute the net mass and the center of mass. The induced error is also second order in the expansion parameter). To simplify the number of issues addressed, we assume for the moment that the particles are more or less homogeneously distributed in a square (Fig.6). In order to make use of the first order approximation, we need some way to divide space. Once such strategy is to introduce a hierarchy of boxes which refine the computational domain into smaller and smaller regions (Fig.7). At refinement level 0, we have the entire computational



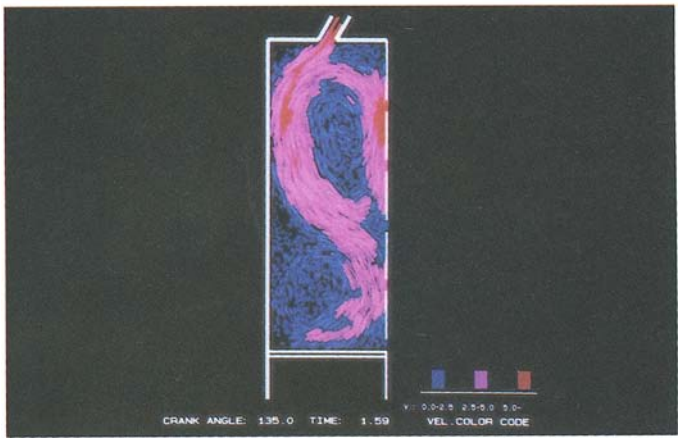
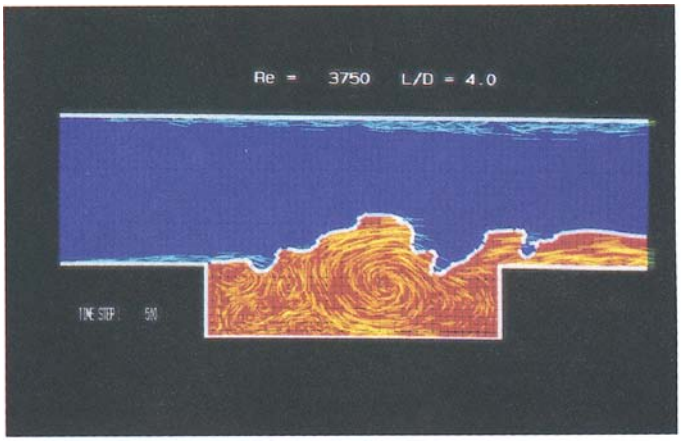
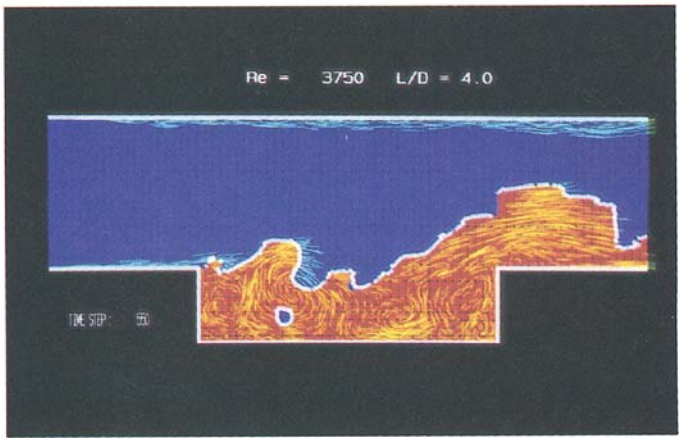


Fig.4: (a) Fluid Motion in a model of an internal combustion engine, by A.F. Ghoniem and L.F. Martins of MIT, during the induction (filling) stroke. As the piston moves away from the cylinder head, the intake port is opened and a gas jet penetrates the cylinder forming a system of large-scale eddies. The figure shows only one half of the cylinder. Each point represents a vortex element and the line is the local velocity vector. The color signifies the absolute value of the velocity.





(b) Combustion in a model of a jet engine, by A.F. Ghoniem and H. Najm of MIT. The flame (white line) is stabilized by the recirculation of the hot combustion products (red) within the chamber cavity. The flow induced oscillations, causing the formation of large-scale eddies, lead to a periodic motion of the flame into the reactants stream (blue). Yellow lines highlight the location and velocity of vortex elements.

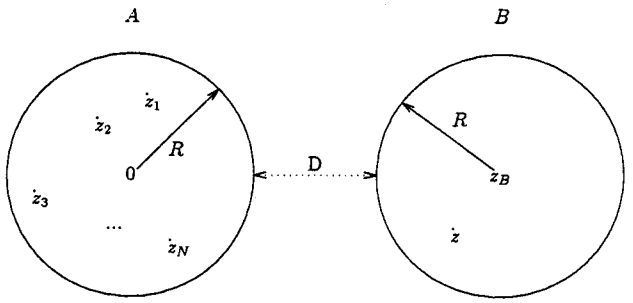


Fig.5: APPROXIMATE CALCULATION OF FORCE. *N* charges are contained in disk *A*, centered at the origin. A multipole expansion, formed about the disk center, can be used to approximate the field at any point outside of *A*. For a point *z* lying inside disk *B*, the error incurred can be determined from the disk radii *R*, the distance separating them *D* and the number of terms used in the expansion.

domain. Refinement level $l + 1$ is obtained recursively from level l by subdivision of each box into four equal parts. This yields a natural tree structure, where the four boxes at level $l + 1$ obtained by subdivision of a box at level l are considered its children.

Definition 0.1 *Two boxes are said to be* **near neighbors** *if they are at the same refinement level and share a boundary point (Fig.8) (A box is a near neighbor of itself).*

Definition 0.2 *Two boxes are said to be* **well separated** *if they are at the same refinement level and are not near neighbors.*

Definition 0.3 *With each box i is associated an* **interaction list,** *which consists of the children of the near neighbors of i's parent which are well separated from box i (Fig.8).*

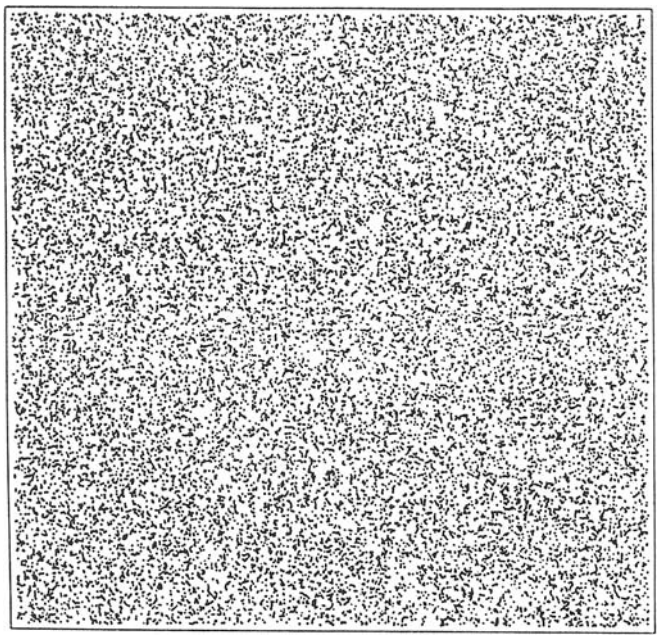


Fig.6: 10,000 charges uniformly distributed in the computational cell.

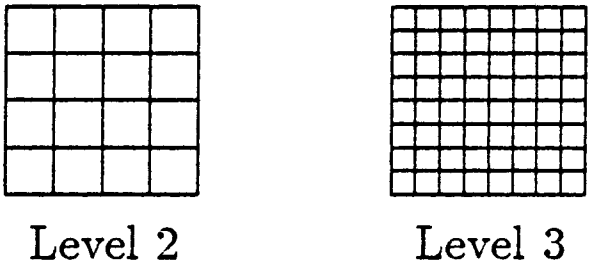


Fig.7: The computational domain shown at the second and third refinement levels. No separation exists between boxes at levels 0 and 1, so that no interactions can be computed by means of expansions.

The basic idea of the method is to create clusters of particles at successive levels of spatial refinement, and to compute interactions between distant clusters by means of the first order approximation. It is clear that at levels 0 and 1, there are no pairs of boxes which are well separated. At level 2, on the other hand, sixteen boxes have been created and there are a number of well separated pairs. The first order approximation can then be used to compute interactions between these well separated pairs. This is the first step of the algorithm. It remains to compute the interactions between particles contained in each box with those contained in the box's near neighbors. The algorithm does this recursively. For example, as we refine each box to create level 3, we seek to determine which other boxes can be interacted with by means of the first order approximation. But notice that those boxes outside the region of the parent's nearest neighbors are already accounted for (at level 2) and that interactions with near neighbors cannot accurately be computed by means of an expansion. The remaining boxes correspond exactly to the interaction list defined above. The nature of

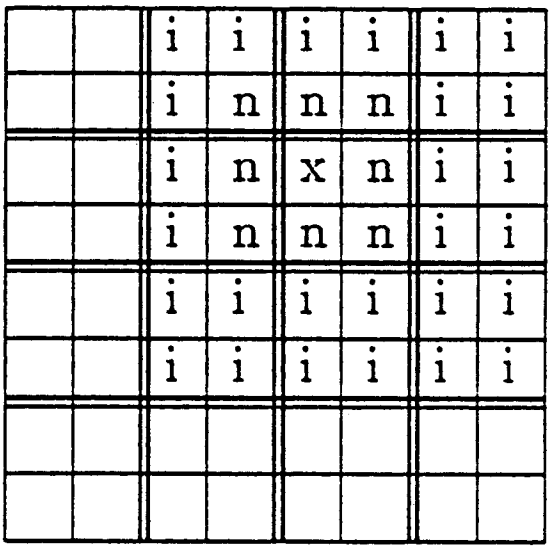


Fig.8: THE NEAR NEIGHBORS AND THE INTERACTION LIST OF A BOX. Double lines correspond to mesh level 2 and thin lines to level 3. The near neighbors of box *x* are those which share a boundary point, and are marked with a *n*. The interaction list for box *x* consists of the children of the near neighbors of *x*'s parent which are separated from *x*. They are marked with an *i*. The size of the interaction list is clearly bounded by 27. (It will be smaller for boxes on the boundary).

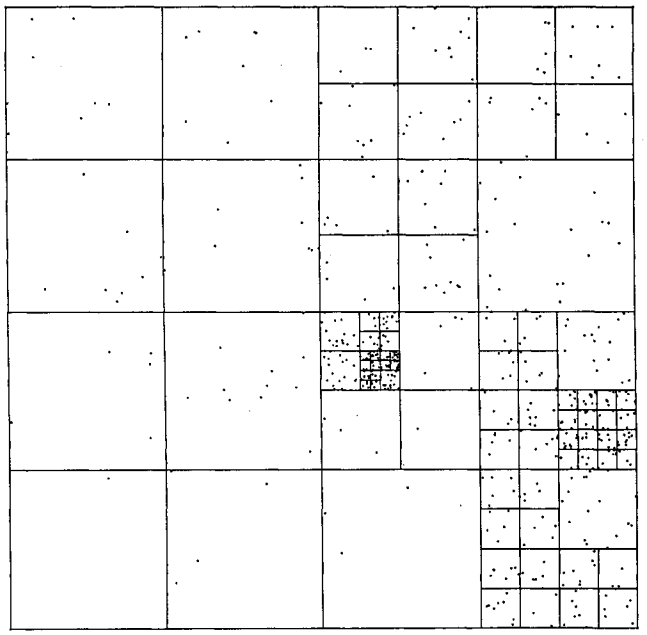


Fig.9: Adaptive refinement of a clustered distribution.

the recursion is now clear. At every level, each box forms the net charge and dipole moment due to the particles it contains. It then evaluates the resulting first order approximation for each particle in the region covered by its interaction list.

We halt the recursive process after $\log N$ levels of refinement. The amount of work done at each level is of the order $O(N)$. To see this, note first that approximately $2 \cdot N$ operations are needed to create all expansions, since each particle contributes to exactly two expansion coefficients. Secondly, from the point of view of a single particle, there are at most 27 boxes (the maximum size of the interaction list) whose expansions are computed, so that $4 \cdot 27 \cdot N$ operations are needed for all evaluations.

At the finest level, we have created $4^{\log_4 N} = N$ boxes and it remains only to compute interactions between nearest neighbors. By the assumption of homogeneity, there is, on the average, one particle per box, so that this last step requires about $8 \cdot N$ operations. The dominant cost is the formation and evaluation of expansions at each level which is $O(N \log N)$.

The Adaptive Algorithm

When the distribution of particles is non-uniform, as in most astrophysical problems, it is clear that a somewhat different strategy must be employed. During the refinement process, each box is examined to determine whether it actually contains any particles. If so, it is subdivided further. If not, it is pruned from the tree structure and ignored at subsequent levels (Fig.9). The complexity of this adaptive algorithm is harder to state precisely, since it depends on the total number of refinement levels which is not determined a priori. For distributions of practical interest, this turns out to be proportional to $\log N$. Note that if the interparticle spacing collapses as N^{-p}, where p is independent of N, then $p \cdot \log N$ levels are needed. It is, therefore, quite reasonable to refer to the adaptive algorithm as also being of the order $N \log N$.

Asymptotic Analysis

The method delineated above will produce results of limited accuracy. In order to increase the precision of the calculation, we obviously want to reduce the error in (8) and (9). This can be accomplished by requiring a greater separation between boxes before the expansions are used, and is one of the strategies employed by the tree codes. Ultimately, this results in a substantial degradation of performance, since more and more interactions are computed directly.

There is another way to improve accuracy, which is to use asymptotic expansions. Rather than stop with the first order approximation to the potential and electric field, we have

$$q_i \cdot \log(z - z_i) = q_i \cdot \log z - \sum_{k=1}^{m} \frac{q_i \cdot z_i^k}{k \cdot z^k} + O\left(\frac{z_i}{z}\right)^{m+1} \tag{10}$$

$$\frac{q_i}{(z - z_i)} = \sum_{k=0}^{m} \frac{q_i \cdot z_i^k}{z^{k+1}} + O\left(\frac{z_i}{z}\right)^{m+1}. \tag{11}$$

The fields due to all charges contained in disk A of Fig.4 can then be represented by full two-dimensional multipole expansions

$$\phi(z) = a_0 \log(z) - \sum_{k=1}^{m} \frac{a_k}{z^k} + O\left(\frac{R}{z}\right)^{m+1} \tag{12}$$

$$E(z) = \sum_{k=0}^{m} \frac{a_k}{z^{k+1}} + O\left(\frac{R}{z}\right)^{m+1}. \tag{13}$$

where

$$a_0 = \sum_{i=1}^{n} q_i \text{ and } a_k = \sum_{i=1}^{n} \frac{q_i \cdot z_i^k}{k} \text{ for } k \geqslant 1. \tag{14}$$

For well separated boxes, the expansion parameter $\frac{R}{z}$ is less than $\frac{1}{2}$. In order to achieve a relative precision of ϵ, we need only choose the number of terms m to be approximately $-\log_2(\epsilon)$. The algorithm of Van Dommelen and Rundensteiner makes use of the full expansion in this manner. The running time remains $O(N \log N)$, but now with a constant of proportionality which depends on the logarithm of the precision. By choosing ϵ to be the machine precision, the fast algorithm is, in fact, slightly more accurate than the direct calculation! Fewer operations are carried out, and less round-off error is introduced. On 32 bit machines, this requires fewer than twenty terms in the expansions.

The Duality Principle

There is a somewhat subtle observation which allows for an algorithm whose running time is proportional to N. For simplicity, we will describe it in the context of the first order approximation. Consider the situation in Fig.10. Suppose that we wish to evaluate the interactions between all groups of particles, without making use of the divide and conquer strategy. With the first order multipole expansion derived above we could do the following:

1. Compute the net charge and dipole moment for each cluster, requiring $2 \cdot n^2$ operations.

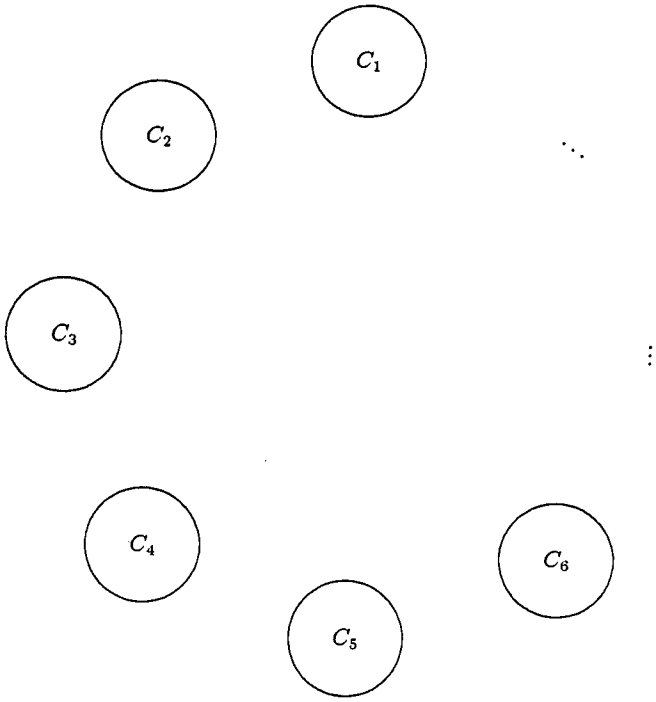


Fig.10: USING THE DUALITY PRINCIPLE. We are given n clusters of particles in the plane, each of which contains n particles. By using both far field (multipole) and local expansions, we can compute all inter-cluster interactions in $O(n^2)$ operations. With multipole expansions alone, the amount of work would be $O(n^3)$, while the direct calculation would require $O(n^4)$ operations.

2. Compute all of the inter-cluster interactions by means of the expansion, requiring $2\cdot(n-1)\cdot n^2$ operations.

The total operation count is of the order n^3.

So far, the approximations we have worked with describe the far field due to a collection of particles. But there is another kind of expansion we can use, which is in some sense the dual of the multipole expansion. In Fig.5, we could expand the field due to the charge q_i located at z_i about the center of disk B.

$$q_i\cdot\log(z-z_i) = q_i\cdot\log((z-z_B)+(z_B-z_i)) \quad (15)$$

$$= q_i\cdot\log(z_B-z_i) + q_i\cdot\frac{z-z_B}{z_B-z_i} + O\left(\frac{z-z_B}{z_B-z_i}\right)^2 \quad (16)$$

$$= \left[q_i\cdot\log(z_B) - \frac{q_i\cdot z_i}{z_B}\right] + (z-z_B)\cdot\left(\frac{q_i}{z_B} + \frac{q_i\cdot z_i}{z_B^2}\right) + O\left(\frac{R}{R+D}\right)^2 \quad (17)$$

In determining the field due to all the charges in disk A, we obtain an expansion about the center of B of the form

$$\phi(z) = b_0 + b_1\cdot(z-z_B) + O\left(\frac{R}{R+D}\right)^2 \quad (18)$$

$$E(z) = \phi'(z) = b_1 + O\left(\frac{R}{R+D}\right)^2, \quad (19)$$

where

$$b_0 = a_0\cdot\log(z_B) - \frac{a_1}{z_B}, \quad (20)$$

$$b_1 = \left(\frac{a_0}{z_B} + \frac{a_1}{z_B^2}\right), \quad (21)$$

and a_0 and a_1 are the net charge and dipole moment of the sources in A. In other words, there is a *local* approximation to $\phi(z)$ which is second order accurate inside B. Moreover, the local expansion coefficients can be computed directly from the far field expansion created for A. There is no need to reexamine the individual particles. In the model problem of Fig.10, we can now use the following strategy:

1. Compute the net charge and dipole moment for each cluster, requiring $2\cdot n^2$ operations.

2. For each pair of clusters, exchange far field coefficients and compute coefficients of the local approximation from (20) and (21), requiring $4\cdot n^2$ operations.

3. Within each cluster, add together all n local approximation coefficients computed in Step 2, requiring $2\cdot n^2$ operations.

4. For every particle, compute the resulting local approximation, requiring $2\cdot n^2$ operations.

By using *two* representations of the fields involved, we have reduced the total operation count from n^3 to n^2.

The Fast Multipole Method

The Fast Multipole Method can be viewed as a tree code, but one which makes use of the duality principle as well as two other analytic observations. For the first observation, suppose that we have created the multipole expansion for each of the four boxes in Fig.11a. We would like to form a single expansion for the parent box without re-examining each particle. The information we have at our disposal consists of the net charge $a_{0_k} = \Sigma_k\, q_i$ and dipole moment $a_{1_k} = \Sigma_k q_i\cdot(z_i - c_k)$ for each child box $k = 1,...,4$, where c_k denotes the child box center. After a little algebraic manipulation, we obtain for the parent box

$$a_0 = \sum_{k=1}^{4} a_{0_k}, \quad (22)$$

and

$$a_1 = \sum_{k=1}^{4} a_{1_k} + a_{0_k}\cdot c_k. \quad (23)$$

In other words, the parent expansion can be computed directly from the coefficients of the four child box expansions.

The second observation is the reverse of this merging procedure. Suppose that we have a local first order approximation expanded about the center of the parent box in Fig.11b. We would like to derive a local expansion for each child box which represents the same approxima-

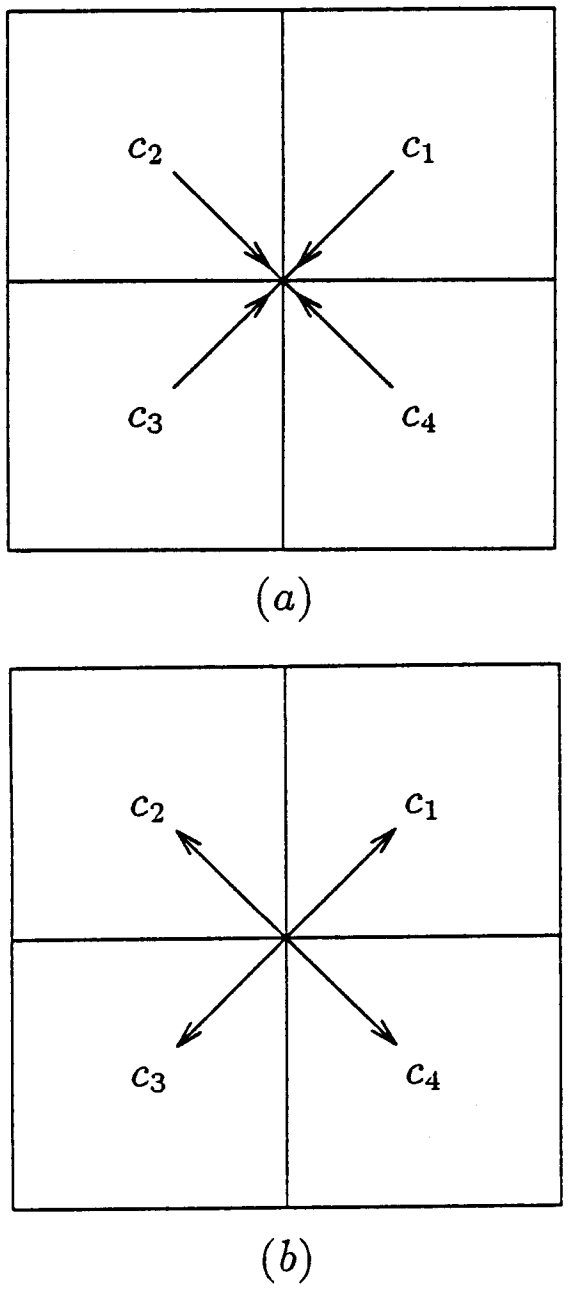


Fig.11: MERGING AND SHIFTING EXPANSIONS. In (a), a single expansion for the parent box is created by merging the four child box expansions. This enables us to form multipole expansions at coarser and coarser levels without re-examining each particle. In (b), a dual procedure is depicted. Given a *local approximation* to the field, valid for the entire parent box, we create four separate expansions, one valid for each child by shifting the expansion center.

tion. This is easy, for if $\Psi(z) = b_0 + b_1 \cdot z$ is the parent's local expansion, then for each child k, we have

$$\Psi_k(z) = (b_0 + b_1 \cdot c_k) + b_1 \cdot (z - c_k). \qquad (24)$$

We have shifted the center of the expansion.

We note now that the previously discussed hierarchical methods were of the order $N \log N$ because all particles were accessed at every level of refinement. The analytic manipulations just described allow us to avoid this, and their incorporation into a tree code results in the Fast Multipole Method. Instead of describing the FMM in detail, we will indicate the modifications which lead to its implementation.

- In the tree code, we proceeded from the coarsest to the finest level, forming multipole expansions for every box. In the FMM, we begin at the finest level, and create multipole expansions from the source positions and strengths. The expansions for all boxes at all higher levels are then formed by the merging procedure delineated above.

- In the tree code, whenever a box b was under consideration, we used its multipole expansion to compute interactions with all particles contained in the boxes of b's interaction list. In the FMM, we *convert* the multipole expansion into a local expansion about the centers of all boxes in b's interaction list.

- After these calculations are completed, we are left with a local expansion in each box at each level. Beginning at the coarsest level, these local expansions are *shifted* to the children's level and added to the children's local expansions. After this recursive process reaches the finest refinement level, a local expansion will have been created for each box which describes the field due to all particles outside the box's near neighbors. It is only this expansion which is evaluated. The near neighbor interactions, as before, are computed directly.

The amount of work done at the finest level is clearly proportional to N. The amount of work done at all coarser levels, however, no longer depends on the number of particles but only on the number of boxes. In a tree of depth $\log N$, there are only $O(N)$ boxes. Hence, the FMM is an $O(N)$ algorithm. In actual implementations, the far field approximation is replaced by a full multipole expansion, and the local approximation is replaced by a power series. In three dimensional calculations, the corresponding expansions are computed in terms of spherical harmonics.

It may not immediately be apparent that much benefit is to be derived from the added complications introduced by the FMM. We suspect that the constant of proportionality of the method can be made substantially smaller than for the other hierarchical codes. This requires a significant amount of analysis concerning the analytic manipulation of expansions[4].

Computer Implementation

The usefulness of the fast algorithms depends, of course, on the point at which they begin to outperform the direct calculation. For small numbers of particles, the overhead in computing expansions outweighs the benefit obtained from their use. In two dimensional calculations, the "crossover point" is in the range of 100-500 particles, depending on the algorithm and the desired precision. In three dimensional calculations, this point is reached somewhat later, when N is between 1000 and 5000. Although early versions of the tree codes and the FMM were implemented on standard sequential machines (such as the VAX), recent experience has shown that they are well-suited to both vector and parallel processors. Hierarchical codes have been written for the Cray-XMP, the Cyber-205, the Connection Machine, the Encore Multimax, and several other machines.

Future Directions

The use of fast hierarchical methods is growing in many of the fields mentioned at the beginning of this article. There are a number of interesting related questions which have yet to be investigated. These range from detailed questions about the best way to incorporate such methods into time-dependent calculations to broader questions about what other classes of problems can be treated by similar means. The development of fast numerical algorithms through the use of asymptotic expansions appears to be in its infancy.

For Further Reading

The hierarchical methods were originally described in the following papers:

A. W. Appel, *An Efficient Program for Many-body Simulation*, Siam. J. Sci. Stat. Comput., 6 (1985), pp. 85-103.

J. Barnes and P. Hut, *A Hierarchical O(N log N) Force-Calculation Algorithm*, Nature, 324 (1986), pp. 446-449.

J. Carrier, L. Greengard, and V. Rokhlin, *A Fast Adaptive Multipole Algorithm for Particle Simulations*, Siam J. Sci. Stat. Comput., to appear (1988).

L. Greengard and V. Rokhlin, *A Fast Algorithm for Particle Simulations*, J. Comput. Phys., 73 (1987), pp. 325-348.

V. Rokhlin, *Rapid Solution of Integral Equations of Classical Potential Theory*, J. Comput. Phys., 60 (1985), pp. 187-207.

L. van Dommelen and E. A. Rundensteiner, *Fast, Adaptive Summation of Point Forces in the Two-Dimensional Poisson Equation*, submitted, J. Comput. Phys., 83 (1989), pp. 126-147.

A more thorough introduction to the Fast Multipole Method, along with much of the necessary mathematics, can be found in:

L. Greengard, *The Rapid Evaluation of Potential Fields in Particle Systems*, MIT Press, Cambridge, 1988.

Work on parallel and supercomputer implementations of the hierarchical methods includes:

J. Barnes, *An Efficient N-body Algorithm for a Fine-grain Parallel Computer*, in The Use of Supercomputers in Stellar Dynamics, (ed. P. Hut and J. McMillan), Springer Verlag, (1986), p. 175.

L. Greengard and W.D. Gropp, *A Parallel Version of the Fast Multipole Method*, in Parallel Processing for Scientific Computing, (ed. G. Rodrigue), SIAM, (1989), p. 213.

L. Hernquist, *Performance Characteristics of Tree Codes*, The Astrophysical Journal Suppl., 64 (1987), p. 715.

J. Makino, *On an O(N log N) Algorithm for the Gravitational N-body Simulation and its Vectorization*, in Proceedings of the First APPI Workshop on Supercomputing, (ed. R. Mendez), Institute for Supercomputer Research Technical Report 8703, (1987), p. 153.

F. Zhao, *On An O(N) Algorithm for Three-dimensional N-body Simulations*, Research Report AI-TR-995, Massachusetts Institute of Technology, (1987).

The application of N-body calculations to astrophysics, plasma physics, and fluid dynamics is discussed in:

S. J. Aarseth, *Dynamical Evolution of Clusters of Galaxies - I*, Mon. Not. R. Astron. Soc. 126 (1963), pp. 223-255.

J. Barnes, *Encounters of Disk/Halo Galaxies.* The Astrophysical Journal, 331 (1988), p. 699.

A.J. Chorin, *Numerical Study of Slightly Viscous Flow*, J. Fluid. Mech., 57 (1973), pp. 785-796.

J.M. Dawson, *Particle Simulation of Plasmas*, Rev. Mod. Phys., 55 (1983), pp. 403-447.

R.W. Hockney and J.W. Eastwood, *Computer Simulation Using Particles*, McGraw-Hill, New York, 1981.

References

1. To improve the accuracy of PM calculations, short-range interactions can be calculated directly, while long-range interactions are obtained from the mesh. Such hybrid algorithms are referred to as particle-particle/particle-mesh (P^3M) methods (see Hockney and Eastwood). A powerful implementation of such ideas in the context of fluid dynamics is given by C.R. Anderson, *A Method of Local Corrections for Computing the Velocity Field Due to a Collection of Vortex Blobs*, J. Comput. Phys., 62, (1986), p.111.
2. See e.g. O.D. Kellogg, *Foundations of Potential Theory*, Dover, New York, 1953.
3. More precisely, the potential is the real part of the indicated expression, and the field is of the form $(Re(\phi'),-Im(\phi'))$.
4. L. Greengard and V. Rokhlin, *On the Efficient Implementation of the Fast Multipole Algorithm*, Technical Report 602, Yale Computer Science Department, 1988.