Parallel indirect multipole BEM analysis of Stokes flow in a multiply connected domain

M.S. Ingber\textsuperscript{a}, A.A. Mammoli\textsuperscript{a} & J.S. Warsa\textsuperscript{b}

\textsuperscript{a}Department of Mechanical Engineering, University of New Mexico, Albuquerque, New Mexico 87131, USA
\textsuperscript{b}Albuquerque High Performance Computing Center, Albuquerque, New Mexico 87131, USA

Abstract

The motion of a statistically significant number of cylinders in a two-dimensional Couette flow is studied using a completed double-layer indirect boundary integral formulation. The condition number of the linear system of equations, comprising the discretized boundary integral equations and the supplementary equilibrium equations, is reduced considerably with the use of a preconditioner based on the structure of the matrix. The speed of the solution of the linear system is increased using a first-shift multipole formulation, in conjunction with an iterative linear equation solver. It is shown that the multipole method is extremely well suited to parallelization, allowing large scale dynamic simulations to be performed in acceptable times. Finally, dynamic simulations are performed to illustrate the behavior of systems with large numbers of particles.

1 Introduction

The boundary element method (BEM) is a very attractive tool for the simulation of particle suspension flows, for a number of reasons. The particle-fluid interactions are modeled exactly, particles of arbitrary shape may be modeled and it is easy to discretize the boundary. Unfortunately, the storage requirement grows as $N^2$, while the calculation time grows at best as $N^2$ in the case of iterative solution,
or $N^3$ in the case of direct solution methods, where $N$ is the number of nodes used to represent the source density. Problems involving a realistic number of particles quickly become too large to handle in acceptable time for even the most powerful supercomputers of today. This is especially true in the case of dynamic simulations.

The multipolar technique has long been in use in the field of astrophysics to model the interactions between large numbers of stars. Barnes and Hut\(^1\) describe an algorithm where the effects of the gravitational fields of stars on a given object star are grouped into increasingly large clusters as the distance from the object star increases. This algorithm results in a total operation count which scales as $N \log N$.

Of particular interest for the simulation of Stokes flow is the recent work of Gòmez and Power\(^2\). There, the multipole method is applied to both direct and indirect BEM formulations. With specified velocity conditions, the indirect approach results in second kind integral equations, which when discretized, produce well-conditioned sets of equations. The solution is in terms of a "hydrodynamic double-layer potential", which has no physical interpretation in fluid flow, except that it may be used in postprocessing to evaluate quantities of interest, such as forces or torques exerted on a particle in the flow.

In the present study, a completed double-layer boundary integral equation is used to represent the model system. The linear system of equations that results from the discretization of the equation is solved iteratively using a Generalized Minimal Residual (GMRES) method. The first-shift multipole technique is used to accelerate the matrix-vector multiplication. To further reduce the operation count, a preconditioner similar to that described by Greenbaum et al.\(^3\) and Greengard et al.\(^4\), based on the structure of the equilibrium equations, is used to render the entire system of equations diagonally dominant.

The multipole method is well-suited to parallelization. The matrix multiplication, which is the process with the largest operation count, can be evenly split between processors. The only communication needed between processors is the gathering of the trial solution vector. All other information is contained within each processor, and does not need to be shared.
2 The completed double-layer boundary integral equation

Consider the two-dimensional flow of fluid at low Reynolds numbers in a bounded domain $\Omega$, containing cylinders of arbitrary cross-section. The boundary is constituted by $M+1$ closed curves, denoted by $L_0$, $L_1$, $L_2$, ..., $L_M$, where $L_0$ denotes the outer boundary. The velocity field satisfies no-slip conditions on each contour $L_p$:

$$u_1(\xi) = U_i(\xi), \xi \in L_p$$ (1)

and the following no-flux conditions:

$$\int_{L_0} U_i(y)n_i(y)dS = \int_L U_i(y)n_i(y)ds$$ (2)

where $L = L_1 \cup L_2 \cup ... \cup L_M$.

The solution of the boundary value problem is sought in terms of a double-layer potential with a density carrying contour $L_0 \cup L$ and unknown density $\phi$:

$$u_i(x) = \int_{L_0 \cup L} K_{ij}(x,y)\phi_j(y)dS.$$ (3)

Double layer potentials do not provide net forces or torques on a closed surface. To account for the force and torque on particles, the equation is 'completed' with $M$ two-dimensional stokeslets with strength $F^p$ and $M$ two dimensional rotlets with strength $T^p$, each pair located in the interior of cylinder $p$, resulting in:

$$u_i(x) = \int_{L_0 \cup L} K_{ij}(x,y)\phi_j(y)dS + \sum_{p=1}^{n} u^i_i(x,y^p)F^p_j + \sum_{p=1}^{n} r^i_j(x,y^p)T^p_j,$$ (4)

where $F^p$ and $T^p$ are the force and torque exerted on particle $p$. The two-dimensional stokeslet is given by:

$$u^i_i(x) = -\frac{1}{4\pi} \left[ \ln \left( \frac{1}{R} \right) \delta_{ij} + \frac{(x_i - y_i)(x_j - y_j)}{R^2} \right],$$ (5)

the rotlet is given by:

$$r^i_j(x) = \frac{\epsilon_{ijk}\delta_{3j}(x_k - y^p_k)}{R^2},$$ (6)
\( y^p \) is the location of the singularities, which must be in the interior of \( L_p \) and \( R = |x - y^p| \). The kernel \( K_{ij} \) is given by:

\[
K_{ij}(\xi, y) = -\frac{1}{\pi} \frac{(\xi_i - y_i)(\xi_j - y_j)(\xi_k - y_k)n_k(y)}{R^4}
\]  

(7)

As the boundary is approached from the interior, Eq. 4 becomes:

\[
U_i(\xi) = \frac{1}{2} \phi_i(\xi) + \int_{L_0} K_{ij}(\xi, y) \phi_j(y) dS + \int_L K_{ij}(\xi, y) \phi_j(y) dS + \sum_{p=1}^{n} u^j_i(\xi, y^p) F^p_j + \sum_{p=1}^{n} r^i_j(\xi, y^p) T^p_j.
\]  

(8)

Note that Eq. 8 is the correct equation for the containing boundary \( L_0 \) and also the particle boundaries, \( L_1 \ldots L_M \).

Uniqueness of the solution requires that an additional constraint be added to the system. This is:

\[
\int_{L_0} \phi_i(y) n_i(y) dS = 0.
\]  

(9)

The incorporation of this constraint into Eq. 8 is described by Gòmez and Power².

Generally the velocity of each particle is not known. Rather, the force and torque applied to each particle are due to gravity or some other well-defined interaction, and therefore known, and the mobility problem is to be solved.

The sum of stokeslet/rotlet pairs becomes part of the RHS. The rigid body motion of the particles may be moved to the LHS of the system, according to the relation:

\[
u(\xi) = u^p + \omega^p \times r^p,
\]  

(10)

where \( r^p = \xi - y^p \), creating an additional \( 3 \times M \) unknowns.

It is shown by Power and Wrobel⁵ that:

\[
F^p_i = \frac{1}{2\pi} \int_{L_p} \phi_j(x) \varphi^i_j(x) dS \text{ for } i = 1, 2 \text{ and } F^p_3 = 0
\]  

(11)

and

\[
T^p_i = 0 \text{ for } i = 1, 2 \text{ and } T^p_3 = \frac{1}{2\pi} \int_{L} \phi_j(x) \varphi^3_j(x) dS,
\]  

(12)

where \( \varphi \) is defined as:

\[
\varphi^i = (\delta_{i1}, \delta_{2i}, 0), \ i = 1, 2 \text{ and } \varphi^3 = (-r^2_3, r^1_3, 0)
\]  

(13)

The required additional set of \( 3 \times M \) equations is therefore provided by applying equilibrium on each particle, using Eqns. 11 and 12.
3 Preconditioning

Using operator notation, the entire system of equations required to solve the mobility problem, comprising the boundary integral equation 8 and the equilibrium equations 11 and 12, may be written concisely as:

\[
\begin{bmatrix}
1/2 + K & U \\
F & 0
\end{bmatrix}
\begin{bmatrix}
\phi \\
u
\end{bmatrix} =
\begin{bmatrix}
b_KU \\
b_F
\end{bmatrix}
\]  (14)

The operator \(1/2 + K\) is diagonally dominant. Deviation from the diagonal is introduced by the integral operator \(K\), and by the submatrices \(U\) and \(F\). A diagonally dominant matrix is well-conditioned, and iterative solvers rapidly converge to an accurate solution. The magnitude of the deviation caused by \(K\) depends on the geometry of the problem, and there is little that can be done to reduce it efficiently. The diagonal dominance reduction effect of the submatrices \(U\) and \(F\) may however be addressed. The preconditioner presented here has the same structure as that described by Greenbaum et al.\(^3\) and Greengard et al.\(^4\). Because of the structure of the matrices in this formulation, the present preconditioner can be computed and applied extremely efficiently. The preconditioner is of the form:

\[
P^{-1} = \begin{bmatrix}
1/2 & U \\
F & 0
\end{bmatrix}^{-1}
\]  (15)

Inversion of \(P\) requires the solution of:

\[
\begin{bmatrix}
1/2 & U \\
F & 0
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2
\end{bmatrix} =
\begin{bmatrix}
b_1 \\
b_2
\end{bmatrix}
\]  (16)

This is done indirectly by first solving for the lower part of the preconditioned vector, \(x_2\):

\[
x_2 = S^{-1}/2(b_2 - 2Fb_1),
\]  (17)

where \(S = -FU\), followed by calculation of \(x_1\) with:

\[
x_1 = 2b_1 - 2Ux_2.
\]  (18)

Since inversion of \(S\) only requires the inversion of \(M\) \(3 \times 3\) blocks (in two dimensions), minimal computational effort is required to apply \(P^{-1}\). In addition, the storage requirements are extremely small.
4 Parallel multipole implementation

Iterative methods are based on the multiplication of a trial solution vector $v$ and the matrix $A$. At each iteration, the residual $b - Av$ is computed and the trial vector $v$ is updated appropriately. The process continues until the residual, or some other related measure of the error in the trial solution, satisfies some convergence tolerance. The matrix-vector multiplication is at the core of iterative methods, and minimizing the operation count needed for its evaluation is crucial for reducing the computer time to solve the linear system.

The multiplication of a row of the coefficient matrix is essentially equivalent to the application of the integral operator $K$ on the trial potential field. The multipole method is based on using Taylor series expansions to cluster far field influences from many boundary elements together, thus allowing the reduction in the operation count associated with performing the boundary integral for each collocation node.

The calculation of the boundary integral for the part of the boundary contained within a cluster consists of the multiplication of moment tensors associated with the cluster (which are only a function of the geometry of the boundary within the cluster, and therefore independent of the location of the collocation node) and the kernel function and its derivatives evaluated at the center of the cluster.

Each collocation node is associated with a list of far field clusters, and a list of near field elements. Every cluster and every element is in turn associated with a set of coefficients used to evaluate the integral over the part of the boundary it represents. The cluster coefficients are common for all collocation nodes, while the element coefficients are not.

Parallelization is implemented at the setup stage, where the coefficients are evaluated, and at the matrix multiplication stage. The setup stage consists of:

1. evaluation of the multipole coefficients for each boundary element - split between processors, coefficients need not be shared,

2. generation of far-field cluster lists and near-field element lists for every node - done locally, results are shared upon completion,

3. calculation of near-field element coefficients - local calculation and storage,
4. evaluation of equilibrium equation coefficients - local calculation and storage,
5. evaluation of preconditioner coefficients - local calculation and storage.

The matrix multiplication is performed in parallel using the following procedure:

1. gather local trial solution vectors into global vector
2. evaluate far field integrals - local
3. evaluate near-field integrals - local
4. apply rigid body motion of particles - local
5. equilibrium equations - local
6. orthogonality conditions - local
7. gather local mapped vectors into global vector
8. apply preconditioner to mapped vector - local
9. gather preconditioned vector
10. distribute global result to local arrays

It is evident from the above that most of the calculation is carried out locally. Inter-processor communication is only required to share cluster list information between processors (step 2 in the setup), and to obtain a complete solution vector for the matrix multiplication process and preconditioning (steps 1, 7 and 9) in the matrix multiplication stage. Gathering of intermediate mapped global vectors is necessary because each part of the matrix, which is essentially divided into blocks of rows, operates on the entire vector.

5 Numerical simulations

The need for fast solutions of linear systems is particularly felt in dynamic simulations, where the evolution of the system configuration in time is approximated by calculating “snapshots” at small time intervals. A problem consisting of 999 identical particles, sized so that
the areal fraction was 0.3, in a Couette viscometer geometry with an inner to outer diameter ratio ($\kappa$) of 0.5, was run on an IBM SP2 parallel computer, with different numbers of processors. Figure 1 shows that the time required per iteration is inversely proportional to the number of processors, if the number of processors is not large. As the number of processors is increased, a constant calculation time of approximately 150 s, which is independent of the number of processors, becomes dominant.

At this point it is not clear whether this constant time is due to communication overhead or to a sequential part of the code. Preliminary results indicate that the latter cause is likely, and code optimization is almost certain to reduce the constant calculation time significantly, allowing full exploitation of parallel processing.

A problem with a smaller number of particles (89) was solved dynamically. The containing geometry was identical to the one used in the larger problem, however the ratio of particle diameter to outer cylinder diameter was now 0.05, maintaining an areal fraction of 0.3. Due to the chaotic nature of systems of large numbers of particles, it is desirable to accurately calculate particle velocities, in order to reduce the influence of numerical error on the particle trajectory. However, error can never be completely eliminated, and it may indeed be useful to study its effect on the evolution of the system.

Two meshes were used, with one mesh twice as refined as the
other. The quantity of most interest here is the particle velocity solution. The exact solution is not available, and an estimate of the error in the velocity calculation may be provided by the difference in results obtained with the two different mesh densities. The maximum differences in particle velocity using the two different meshes, normalized by the maximum velocity vector magnitude, were 1.6%, 1.7% and 12.3% in the \( x \), \( y \) and \( \theta \) directions respectively. The corresponding average differences were 0.2%, 0.3% and 1.8%. Notice that the difference is largest in the rotational velocity component. This is a characteristic of the indirect mobility formulation.

While the average difference is acceptable, the maximum difference is high, especially in the rotational velocity component. The need for adaptive localized mesh refinement is evident, especially in dynamic simulations, where particles will approach each other and the containing boundary down to a specified minimum separation. The particle positions after 1/2 revolution of the inner cylinder are shown in Fig. 2. While the initial distribution was random, particles are now arranged in chain-like structures. The most important feature, however, is that the positions predicted with the fine and coarse mesh are visibly different, even after only 1/2 revolution. This is especially true in areas where multiple-particle interactions have occurred. This is a clear indication of the strong non-linearity of the system, and of the need to minimize the error in the particle velocity calculations.

The average particle radial position, as predicted using the fine and coarse meshes, is plotted in Fig. 3. In both cases, the average radial position appears to increase with time, however the rate of increase is much larger in the case of the coarse mesh. From this comparison, it appears that error in the velocity prediction gives rise to net particle migration, increasing in rate as the error increases.

Finally, the number of GMRES iterations required to solve the 89 particle problem and the 999 particle problem were identical. This is an important result. A constant number of iterations is essential to maintain the \( N \log N \) scaling of the first shift formulation, or the \( N \) scaling of the second shift formulation.
Figure 2: Particle positions after 1/2 revolution for simulations with 1180 elements (left) and 595 elements (right). Particles are separated by at least 1/100 of a radius to avoid singularities.

6 Conclusions

The most important conclusion to be drawn from this study is that very efficient parallelization is achievable with the multipole BEM implementation. Load balancing can be achieved easily by effectively dividing the number of matrix rows as evenly as possible between processors. The number of GMRES iterations appears to be independent of $N$, so that the total solution time is only a function of the matrix multiplication operation count.

Although the time required per iteration asymptotes to $N \log N$ scaling, the reduction in the operation count only becomes evident for large problems. This is because the number of domain subdivisions must be kept small compared to the number of elements, resulting in a relatively large number of elements in the near field. Assuming a constant, large number of elements in the near-field, the number of far-field elements must be much larger to achieve noticeable efficiency gains.

Finally, it is shown that, in order to obtain accurate solutions, mesh refinement in areas of near singularity must be applied. Significant errors arise even when the distance of the collocation point from the boundary is of the order of 10% of the element size. The behavior of multiple-body systems is known to be chaotic. It is shown here
that relatively small errors may lead to large changes in results. It also appears that the error leads to particle migration, in this case outward. By using dynamic mesh refinement and variable mapping in the integration routines, the error intrinsic in the BEM calculation can be minimized, and a controlled error may be introduced. This would allow the study of the influence of error on the macroscopic behaviour of the system. This type of study represents a typical application of the multiple-body simulations presented here.

Acknowledgment

This research was supported by the U.S. DOE grants DE-FG03-97ER25332 and DE-FG03-97ER14778/A0. The financial support does not constitute an endorsement by the DOE of the views expressed in this article. Additional research support was provided by the Maui High Performance Computing Center under cooperative agreement number f29601-93-2-0001 from the Air Force Research Laboratory.
References


