Chapter 19

Three-Dimensional Numerical Results

There are three objectives behind the numerical examples (mostly drawn from the work of Karrila *et al.* [39] and Fuentes [20]) presented in this chapter. First, we test the CDL-BIEM concept itself, by comparing these results with those obtained from other well-established analytical, numerical, and experimental results. These tests include a selection of mobility and resistance relations for one-, two-, and three-sphere geometries and sedimentation experiments for regular polyhedra. Second, we examine the iterative algorithm to demonstrate that the solution obtained is identical numerically to that obtained by Gaussian elimination and to show that the rate of convergence is consistent with that predicted by a spectral analysis of the continuous equations. For all cases in which the spectral radius is known, we will see that the method converges according to these estimates. Finally, we test the multisphere mobility algorithm on a parallel computer (Sequent Symmetry). The results indicate that computational loads can be balanced quite evenly in a multiple-processor environment, and speedups close to the ideal limit are achieved.

The discussions in Chapters 15 through 17 emphasized the continuous equations, and the conclusions reached in those sections are independent of the choice of the discretization scheme. Actual numerical results are of course dependent on the discretization method. The results presented here are all for triangular boundary elements with characteristic functions as the basis. While relatively large numbers of boundary elements are required, our main objective here is to demonstrate the behavior of some discretized version of CDL-BIEM for a representative sample of three-dimensional structures. The number of elements can be reduced substantially by the use of more elaborate (higher order) basis elements [6, 11], but a price must be paid in the form of more elaborate pre- and post-processing steps, especially in connection with the construction of parallel algorithms and the Riesz–Lorentz construction of the surface tractions. Other approaches to discretizations are possible, for example, global collocation over the surface, as shown in the previous chapter and the original literature (Karrila and Kim [38]). Global collocation is competitive for simple shapes, such as spheres, and in axisymmetric geometries. However, for more

complex structures, the simple discretization may be preferable, not only on the basis of generality, but also because of the low computational cost of vectorized operations performed as parallel tasks.

19.1 Discretization with Constant Elements

Our example centers on discretization of the following boundary integral equation for the mobility of a single particle.

$$\varphi(\boldsymbol{x}) + \mathcal{K}\varphi(\boldsymbol{x}) + \sum_{\nu=1}^{6} \varphi^{(\nu)} \langle\, \varphi^{(\nu)}, \varphi \,\rangle = \left(\boldsymbol{F} - \frac{1}{2}\boldsymbol{T} \times \nabla\right) \cdot \frac{\mathcal{G}(\boldsymbol{x} - \boldsymbol{x}_c)}{8\pi\mu} , \qquad (19.1)$$

where $\boldsymbol{F}$ and $\boldsymbol{T}$ are the hydrodynamic forces and torques exerted by the fluid on the particle, and $\varphi^{(\nu)}$, $\nu = 1, 2, \ldots, 6$ are the null solutions given explicitly by

$$\begin{aligned}
\varphi^{(1)} &= \frac{e_1}{\sqrt{A}} , && \text{RBM translation in the } x\text{-direction} \\
\varphi^{(2)} &= \frac{e_2}{\sqrt{A}} , && \text{RBM translation in the } y\text{-direction} \\
\varphi^{(3)} &= \frac{e_3}{\sqrt{A}} , && \text{RBM translation in the } z\text{-direction} \\
\varphi^{(4)} &= \frac{x_2 e_3 - x_3 e_2}{\sqrt{I_1}} , && \text{RBM rotation about the } x\text{-axis} \\
\varphi^{(5)} &= \frac{x_3 e_1 - x_1 e_3}{\sqrt{I_2}} , && \text{RBM rotation about the } y\text{-axis} \\
\varphi^{(6)} &= \frac{x_1 e_2 - x_2 e_1}{\sqrt{I_3}} . && \text{RBM rotation about the } z\text{-axis}
\end{aligned}$$

Here, the origin and directions of the coordinate axes are taken as the "particle shell" center of mass and the principal directions of the "particle shell" moment of inertia tensor. With the above scaling, where A is the particle surface area and I_1, I_2, and I_3 are the moments of inertia about the principal axes, *i.e.*,

$$I_1 = \int_S (y^2 + z^2)\, dS , \quad I_2 = \int_S (z^2 + x^2)\, dS , \quad I_3 = \int_S (x^2 + y^2)\, dS ,$$

the densities are orthonormal with respect to the natural inner product, *i.e.*,

$$\left\langle \varphi^{(\mu)}, \varphi^{(\nu)} \right\rangle = \delta_{\mu\nu} .$$

For the unit sphere, $I_1 = I_2 = I_3 = 8\pi/3$.

Now suppose that the surface is divided into N boundary elements. The exact shape of these elements is not important in the subsequent discussion, although in the codes discussed later on, the elements are triangular. We identify these N elements with the integer label j, with $1 \le j \le N$. We approximate φ as piece-wise constant, so that on each element (say element j) the x, y, and z components of the surface density φ constitute three unknowns, for a total of

$3N$ unknowns. These we denote by $\varphi_\alpha(j)$, ($\alpha = 1,2,3$; and $1 \leq j \leq N$). We may adopt the convention of ordering these unknowns into a vector array as follows:

$$\left(\begin{array}{c} \varphi_1(1) \\ \vdots \\ \varphi_1(N) \\ \hline \varphi_2(1) \\ \vdots \\ \varphi_2(N) \\ \hline \varphi_3(1) \\ \vdots \\ \varphi_3(N) \end{array}\right),$$

that is, we load all the x-components, then all the y-components, then all the z-components.[1]

Fortunately we also have $3N$ equations, since on each boundary element there are three equations obtained from the boundary conditions for v_x, v_y, and v_z. Following the same ordering convention as was used with the unknowns, we order the $3N$ equations as follows:

the N x-equations for elements 1 to N
the N y-equations for elements 1 to N
the N z-equations for elements 1 to N .

An example program, **CDLBIEM.FOR**, is available over the Internet network, and the rest of this section will be easier to follow with it in hand. The procedure for obtaining this from the *anonymous* user account on "Flossie" is as described in the beginning of this book.

```
% ftp flossie.che.wisc.edu
Name: anonymous
Anonymous user OK. Enter real ident... yourname
* cd chapter19
* get cdlbiem.for
* bye
```

We consider how each term in Equation 19.1 loads into the system matrix. The first term on the LHS simply leads to the 3×3 block matrix,

$$\left(\begin{array}{c|c|c} I_N & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & I_N & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} & I_N \end{array}\right),$$

[1] In some instances, *e.g.*, for parallel computer architectures, it may be more efficient to order the unknowns in triads associated with each boundary element.

with $N \times N$ identity matrix along the block diagonal.

This double layer term requires a little more work. We write this approximately as

$$\mathcal{K}\varphi \;=\; \int_{S(y)} \boldsymbol{K}(\boldsymbol{x},\boldsymbol{y}) \cdot \varphi(\boldsymbol{y})\, dS(\boldsymbol{y}) \;\sim\; \sum_{j=1}^{N} [\boldsymbol{K}(\boldsymbol{x},\boldsymbol{y}(j)) \cdot \varphi(\boldsymbol{y}(j))]\, \Delta A(j)\ ,$$

where $\Delta A(j)$ is the area of the j-th boundary element. The kernel $\boldsymbol{K}(\boldsymbol{x},\boldsymbol{y})$ is given by

$$\boldsymbol{K}(\boldsymbol{x},\boldsymbol{y}) \;=\; \frac{3}{2\pi} \frac{\boldsymbol{n}(\boldsymbol{y}) \cdot (\boldsymbol{x}-\boldsymbol{y})}{|\boldsymbol{x}-\boldsymbol{y}|^3} \frac{(\boldsymbol{x}-\boldsymbol{y})}{|\boldsymbol{x}-\boldsymbol{y}|} \frac{(\boldsymbol{x}-\boldsymbol{y})}{|\boldsymbol{x}-\boldsymbol{y}|}\ .$$

Note that in the program listing **CDLBIEM**, $\boldsymbol{y}$ (as `y()`) is also used as the "dummy variable" of integration. In what follows, we should keep in mind that we are setting the boundary condition on element i, while focusing our attention on the contribution from boundary element j. This, of course, leads to the i,j element for each of the nine submatrices in the 3×3 block matrix.

The integrand, $\boldsymbol{K}(\boldsymbol{x},\boldsymbol{y}) \cdot \varphi$, is

$$\frac{3}{2\pi} \frac{\boldsymbol{n}(\boldsymbol{y}) \cdot (\boldsymbol{x}-\boldsymbol{y})}{|\boldsymbol{x}-\boldsymbol{y}|^3} \left(\frac{(x_\beta - y_\beta)}{|\boldsymbol{x}-\boldsymbol{y}|} \cdot \varphi_\beta \right) \frac{(\boldsymbol{x}-\boldsymbol{y})}{|\boldsymbol{x}-\boldsymbol{y}|}\ ,$$

and if the block matrix is written as

$$\left(\begin{array}{c|c|c} A^{(11)} & A^{(12)} & A^{(13)} \\ \hline A^{(21)} & A^{(22)} & A^{(23)} \\ \hline A^{(31)} & A^{(32)} & A^{(33)} \end{array} \right)\ ,$$

we find that

$$A_{ij}^{(\alpha\beta)} = \frac{3}{2\pi} \frac{\boldsymbol{n}(\boldsymbol{y}(j)) \cdot (\boldsymbol{x}(i)-\boldsymbol{y}(j))}{|\boldsymbol{x}(i)-\boldsymbol{y}(j)|^3} \frac{(x_\alpha(i) - y_\alpha(j))}{|\boldsymbol{x}(i)-\boldsymbol{y}(j)|} \frac{(x_\beta(i) - y_\beta(j))}{|\boldsymbol{x}(i)-\boldsymbol{y}(j)|}\ .$$

Note that if $i = j$, that is, if we look at the contribution from the boundary element at which we are evaluating the boundary condition, the kernel vanishes identically, since

$$\lim_{\boldsymbol{y}\to\boldsymbol{x}} \boldsymbol{n}(\boldsymbol{y}) \cdot (\boldsymbol{x}-\boldsymbol{y}) \;=\; \boldsymbol{0}$$

on a Lyapunov surface. This simplification does not occur in the single layer distribution, and a special set of instructions must be included in the code to calculate the contribution from this element [72]. For CDL-BIEM with nonplanar elements and/or more sophisticated basis functions (and quadratures), this contribution to the integral also requires special logic in the codes, resulting in a corresponding degradation of the parallel algorithm (see Exercise 19.1 for an estimate of the contribution from a "polar cap" on a sphere).

We now consider the Wielandt deflation terms associated with particle translations. With $\nu = 1,2,3$, we write

$$\varphi^{(\nu)} \langle\, \varphi^{(\nu)}, \varphi \,\rangle \;=\; \frac{e_\nu}{A} \int_S e_\nu \cdot \varphi(y)\, dS(y) \;=\; \frac{e_\nu}{A} \int_S \varphi_\nu(y)\, dS(y)$$
$$\sim\; \frac{e_\nu}{A} \sum_{j=1}^{N} \varphi_\nu(j) \Delta A(j)\,.$$

Thus for $\nu = 1,2,3$ the Wielandt deflation terms contribute the following 3×3 block diagonal matrix:

$$\frac{1}{A} \left(\begin{array}{c|c|c} diag(\Delta A(j)) & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & diag(\Delta A(j)) & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} & diag(\Delta A(j)) \end{array} \right) .$$

Now consider the deflations associated with the rotational motions. For $\nu = 4$, the associated null solution is a rotation about the x-axis, and we write

$$\varphi^{(4)} \langle\, \varphi^{(4)}, \varphi \,\rangle \;=\; \frac{(x_2 e_3 - x_3 e_2)}{I_1} \int_S (y_2 \varphi_3 - y_3 \varphi_2)\, dS(y)\,.$$

Thus for $\nu = 4$, the Wielandt deflation terms contribute the following 3×3 block matrix:

$$\left(\begin{array}{c|c|c} \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & \frac{x_3(i) y_3(j) \Delta A(j)}{I_1} & -\frac{x_3(i) y_2(j) \Delta A(j)}{I_1} \\ \hline \mathbf{0} & -\frac{x_2(i) y_3(j) \Delta A(j)}{I_1} & \frac{x_2(i) y_2(j) \Delta A(j)}{I_1} \end{array} \right) .$$

The final result is obtained by including the contributions from the cases with $\nu = 5$ and $\nu = 6$, which can be obtained by cycling the indices $1,2,3$, wherever they appear in the above matrix (including the matrix element labels).

The RHS vector is obtained simply by evaluating the Stokeslet and rotlet at the i-th boundary element, as can be seen from the corresponding lines (147–181) in the listing.

The *deflation at the other end of the spectrum* is essential for rapid convergence of the iteration scheme. The eigenvalue, $\lambda = +1$, of $\mathcal{K}$ is deflated by using the eigenfunction of the adjoint, $\mathcal{K}^*$. Recall that this eigenfunction is simply the surface normal. The important thing to remember is that the system $(I + \mathcal{K})x = b$ has the same solution as

$$\left(I + \mathcal{K} - \frac{vv^t}{|v|^2} \right) x \;=\; \left(I - \frac{vv^t}{\lambda |v|^2} \right) b \;=\; \tilde{b}\,,$$

which (with $\lambda = 2$ for the operator $I + \mathcal{K}$) leads to the iterative scheme,

$$x_{n+1} \;=\; \left(-K + \frac{vv^t}{|v|^2} \right) x_n \;+\; \tilde{b} \;=\; -K x_n + \frac{v \langle v, x_n \rangle}{|v|^2} \;+\; \tilde{b}\,.$$

Number of elements	$8\pi\mu aU/F$
320	1.3594
1280	1.3370
5120	1.3329

Table 19.1: Effect of mesh refinement on sphere mobility (exact: 4/3).

In our problem, the inner product terms are simply

$$\frac{v\,\langle v, x_n\rangle}{|v|^2} \rightarrow \frac{\boldsymbol{n}(\boldsymbol{x})}{A}\int_S \boldsymbol{n}(\boldsymbol{y})\cdot\boldsymbol{\varphi}(\boldsymbol{y})\,dS(\boldsymbol{y}) \sim \frac{\boldsymbol{n}(\boldsymbol{x})}{A}\sum_{j=1}^{N}(\boldsymbol{n}(\boldsymbol{y})\cdot\boldsymbol{\varphi}(\boldsymbol{y}))\Delta A(j)$$

$$\frac{v\,\langle v, b\rangle}{2|v|^2} \sim \frac{\boldsymbol{n}(\boldsymbol{x})}{A}\sum_{j=1}^{N}(\boldsymbol{n}(\boldsymbol{y}(j))\cdot b)\Delta A(j)\,,$$

and these are worked out in lines 196–205 (for $\langle v, b\rangle$) and 226–236 (for $\langle v, x_n\rangle$). Note that $\tilde{b}$ is designated as `bt()` in the code.

A final note: The listing **CDLBIEM** is written as if we are solving $Ax = b$ by iterating $x_{n+1} = x_n - Ax_n + b$. But the matrix is of the form $A = I + \mathcal{K}$, and in its construction, we explicitly added the identity (lines 137–145). The program should be streamlined by deleting lines 137–145 and the terms `xold()` in lines 247–252.

19.2 Resistance and Mobility of Spheres

In this section, we examine results for the resistance and mobility functions obtained by the CDLBIEM code discussed in the preceding section. We proceed with a convenient tessellation of the sphere surface: Start with an icosahedron and get subsequent mesh refinement by subdividing each triangular element into four triangular subelements repeatedly. As expected, discretization error diminishes with mesh refinement (see Table 19.1). The 320-element polyhedron reproduced the Stokes law drag with a relative error of about 1%. As an even more stringent test, we can compute the spectrum numerically and compare with the analytical result. The lower modes should be reproduced faithfully by the discretized system. In fact, as in many other related problems (*e.g.*, simulation of the frequencies of a stretched membrane with finite elements), the highest modes of the discretized system cannot be trusted, but fortunately the lowest modes contain the information of interest. Discretization of compact operators introduces large errors only for the higher modes that could not be resolved by the chosen discrete mesh. Our results for the lower modes agreed with the analytical results of Chapter 17 to within 1% as well (see Figure 17.4 in Chapter 17 for a typical example).

In Figures 19.1–19.3, we compare CDL-BIEM with other methods of computing the mobility functions for two spheres. Figure 19.1 shows the sphere mobilities as functions of the sphere-sphere separation for an external force on sphere 1, acting along the line of centers, with the second sphere force-free. This problem is described by the mobility relations (see Chapter 11),

$$6\pi\mu a U_1 = x^a_{11} F_1 \tag{19.2}$$
$$6\pi\mu a U_2 = x^a_{21} F_1 \,, \tag{19.3}$$

with all forces and motions along the line of centers. The solid curves are the exact results obtained by using bispherical coordinates. Figures 19.2 (translational velocities) and 19.3 (rotational velocities) show results for an external force on sphere 1, acting normal to the line of centers, with the second sphere force-free, as described by the mobility relations

$$6\pi\eta a \boldsymbol{U}_1 = y^a_{11} \boldsymbol{F}_1 \tag{19.4}$$
$$6\pi\eta a \boldsymbol{U}_2 = y^a_{21} \boldsymbol{F}_1 \tag{19.5}$$
$$4\pi\eta a^2 \boldsymbol{\omega}_1 = y^b_{11} \boldsymbol{F}_1 \times \boldsymbol{d} \tag{19.6}$$
$$4\pi\eta a^2 \boldsymbol{\omega}_2 = y^b_{21} \boldsymbol{F}_1 \times \boldsymbol{d} \,, \tag{19.7}$$

with all translation along the direction of the external force and the rotations about an axis orthogonal to both the line-of-centers and the direction of the external force. The solid curves are the boundary collocation results from [41] (see also Chapter 13 and the program listing, **mandr.for**).

The agreement between CDL-BIEM and the values in the literature is excellent, for all three cases, except for almost-touching spheres. The small (less than 1%) relative errors may be attributed to discretization errors, which may be eliminated with further mesh refinement. For almost-touching spheres, the meshes must resolve the gap geometry, and this may be accomplished by refining the mesh in the gap region.

Figure 19.4 shows results for three fixed spheres placed in an equilateral-triangle configuration. Our results for the drag are in good agreement with the multipole solution presented in Kim [42] over a wide range of sphere-sphere separations. If we extrapolate the CDL-BIEM results, we obtain an estimate for a cluster formed by three touching spheres, which is in good agreement with the experimental result obtained by Lasso and Weidman [44]. We should expect relative errors of less than 2%, so this better-than-expected agreement should be attributed to cancellation of errors.

19.3 Sedimentation of Platonic Solids

The robustness of our simple collocation scheme (planar triangular elements and characteristic functions as basis elements) is demonstrated in Figure 19.5 with drag-coefficient calculations for regular polyhedra. These shapes are quite commonly encountered in suspensions because of the crystal structure of the

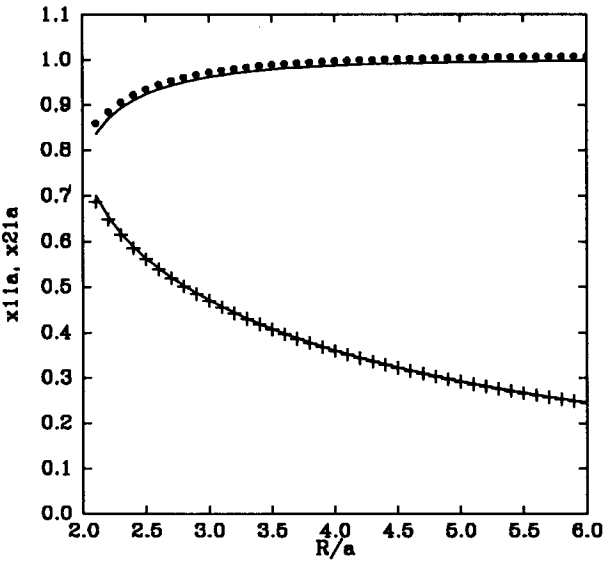

Figure 19.1: Two-sphere mobilities x^a_{11} (•) and x^a_{21} (+) *vs.* the separation (R/a); solid curves (exact solution).

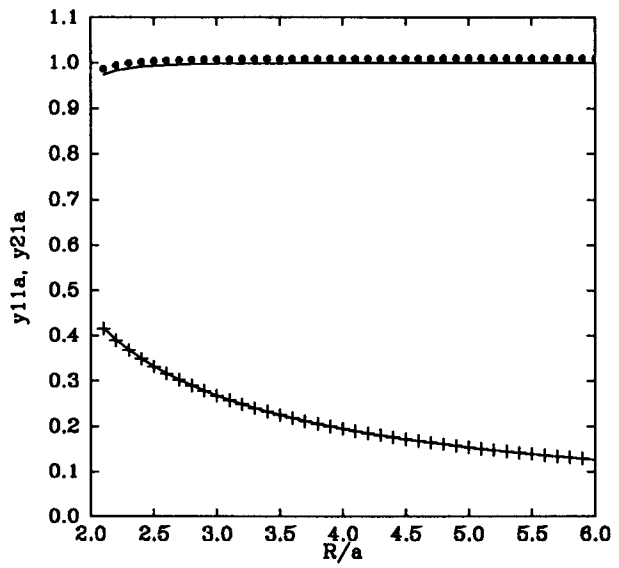

Figure 19.2: Two-sphere mobilities y^a_{11} (•) and y^a_{21} (+) *vs.* the separation (R/a); solid curves from Kim and Mifflin.

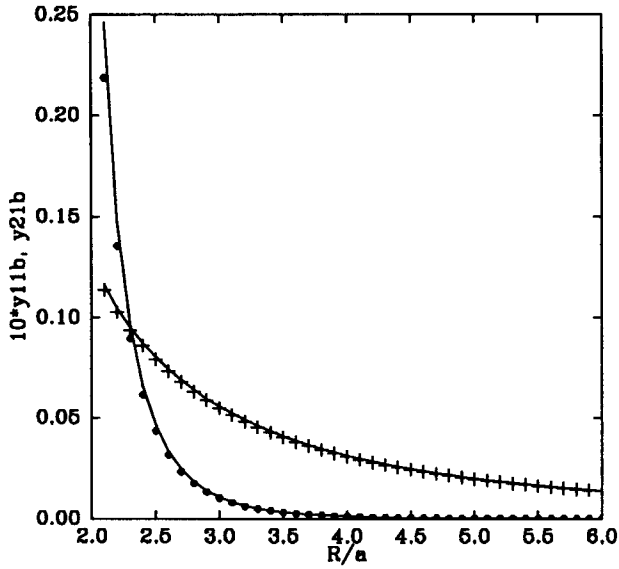

Figure 19.3: Two-sphere mobilities y_{11}^b (•) and y_{21}^b (+) *vs.* the separation (R/a); solid curves from Kim and Mifflin.

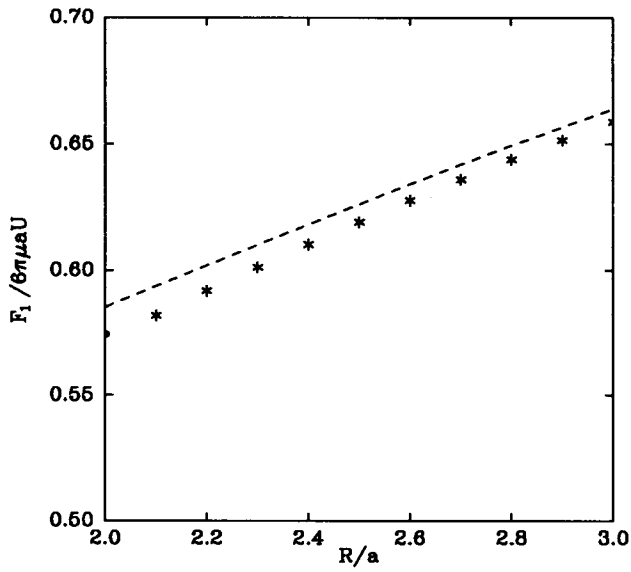

Figure 19.4: Drag on a sphere, for three spheres in an equilateral-triangle configuration: CDL-BIEM (∗); analytical (—), and experimental (•).

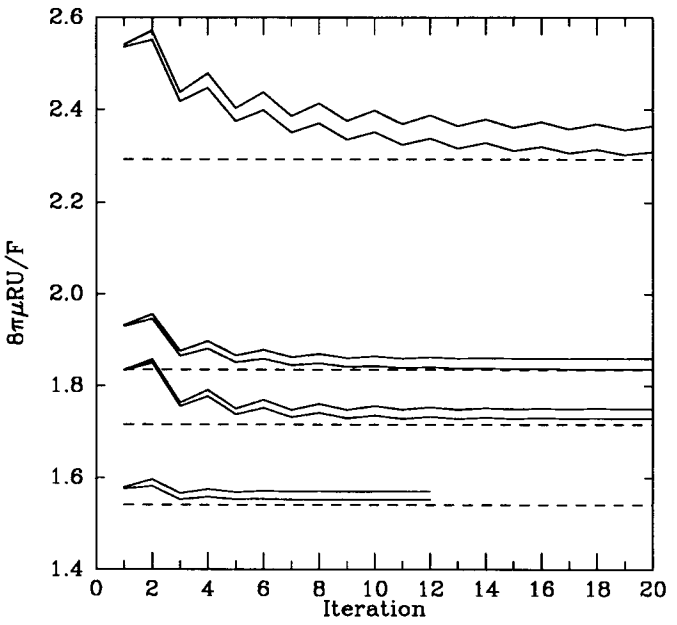

Figure 19.5: Convergence of iterations, at various discretizations. From top to bottom: tetrahedron, octahedron, cube, and icosahedron. Experiments of Pettyjohn and Christiansen (dashed lines). R is radius of circumscribed sphere.

mineral constitutents. For these shapes, the collocation equation is further simplified because, as explained earlier, *elements on the same polyhderal face* do not interact. Figure 19.5 gives the sedimentation velocities for a tetrahedron (256 and 1024 elements), octahedron (512 and 2048 elements), cube (268 and 1072 elements), and icosahedron (320 and 1280 elements). The number of iterations required to obtain convergence and the improvement obtained with mesh refinement are also shown. These results are in good agreement with the experiments of Pettyjohn and Christiansen [56], as represented in the figure (horizontal dashed lines).

As in Karrila and Kim [38] we find that CDL-BIEM works even when sharp corners (which are not allowed in the Fredholm–Riesz–Schauder theory) are present. We can explain this with heuristic reasoning based on the energy dissipation (inclusion monotonicity) theorems of Hill and Power [30]. If we round the corners by an infinitesimal amount, the drag changes only slightly, but now the Fredholm–Riesz–Schauder theory is applicable. Near the corners, we have steep velocity gradients and therefore large energy dissipation rates. These, in turn, imply eigenvalues near ± 1, and thus a slower rate of convergence for the deflated system. As expected, this effect is most pronounced for the tetrahedron and least evident for the icosahedron. However, even for the tetrahedron, the simple scheme still provides reasonable results.

Method	MicroVAX II	Astronautics ZS-1
LU Factorization	6150	360
Direct Iteration	140	8

Table 19.2: Computational times (seconds) for the 320-element mesh for the single sphere: comparison of LU factorization and iteration solutions.

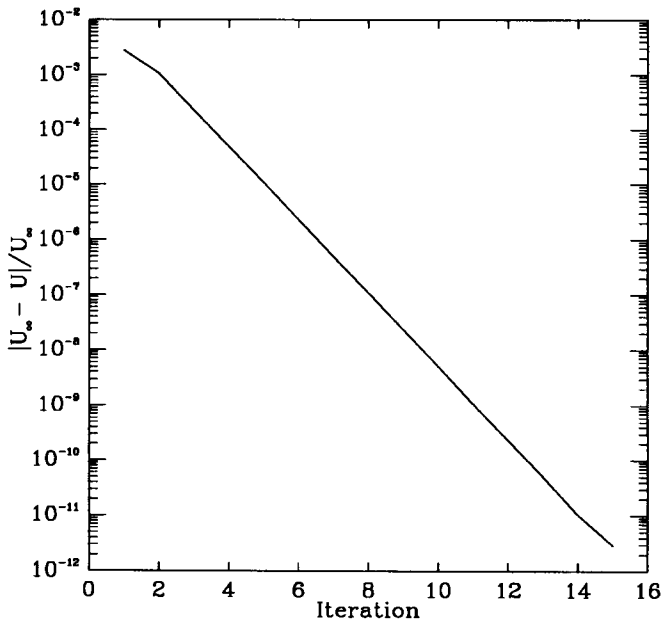

Figure 19.6: Convergence of deflated mobility problem (320 elements).

19.4 Benchmarks

For the purpose of comparison, the 320-element, single-sphere problem was solved by LU factorization (with the LINPACK routines DGECO and DGESL) and direct (Picard) iteration. Computation times are given in Table 19.2 for both the Micro VAX II and the Astronautics ZS-1. For this particular problem, the iterative scheme was roughly 45 times faster than LU factorization. The convergence of the iterative solution is shown in Figure 19.6.

The two-sphere problem converged to five significant figures after 16 iterations, with each iteration taking 6.1 seconds on the ZS-1 (in double precision). As expected, each iteration took approximately four times longer than the iterations in the single-sphere problem. Convergence rates of the two-sphere iterations are shown in Figures 19.7 and 19.8 for spheres with center-to-center

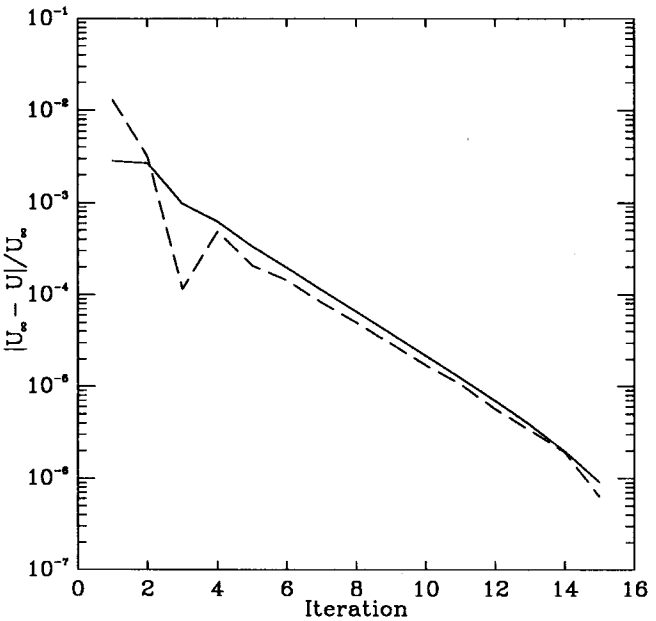

Figure 19.7: Convergence of two-sphere translational mobilities (320 elements, $R/a = 2.5$): sphere 1 (solid line), sphere 2 (dashed line).

separations $R = 2.5a$. In all cases, we used for the initial guess, $\varphi^{(0)}(\xi) = 1$. The important point here is that the rate of convergence is very fast, because the spectral radius is approximately 0.6 (see Chapter 17).

19.5 CDL-BIEM and Parallel Processing

Technological advances may increase the speed of the fastest processors by a factor of ten in the future. Ultimately, these improvements are limited by two fundamental constraints: the speed of light (which imposes an upper bound on the movement of information from one part of the computer to another) and the size of an atom (which imposes a lower bound on processor dimensions). Together, they represent a theoretical upper bound on the computational speed of a single processor.

Parallel processing offers a way of skirting these limitations, if the problem at hand can be broken into many parallel streams that can be tackled by an array of processors working independently. For problems in which this simple idea works, phenomenal gains in computational speed can be envisioned. For CDL-BIEM, the entire step, from matrix construction to Wielandt deflation to iterative solution, can be performed in parallel. Therefore, we digress here briefly to consider the current situation in parallel architectures. *The main objective of this discussion is to illustrate the point that the details of an opti-*

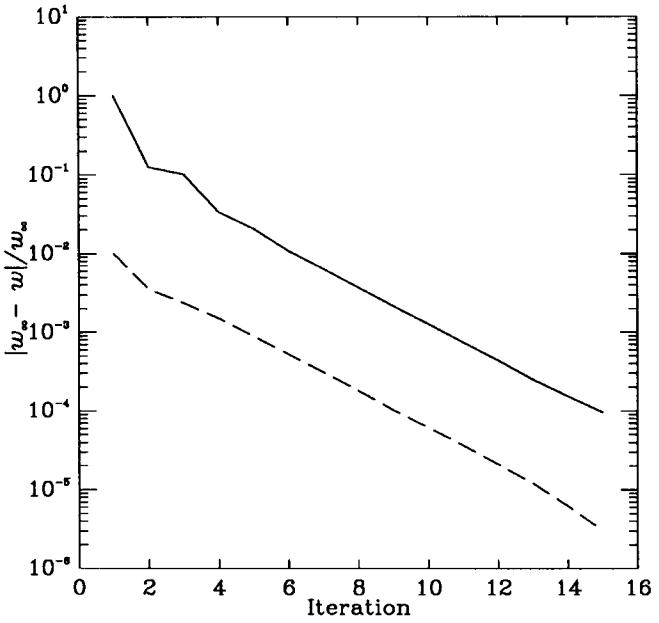

Figure 19.8: Convergence of two-sphere rotational mobilities (320 elements, $R/a = 2.5$): sphere 1 (solid line), sphere 2 (dashed line).

mized CDL-BIEM code will vary with computer architecture. For the forseeable future, developments in algorithms and architectures are coupled. Naturally, the scope of the discussion in this book must be restricted to general mathematical statements that are independent of architecture, but that form the basis for more detailed research for a specific computer architecture of interest. A perspective on the relation between architecture and the operations of numerical linear algebra may be gained from a number of sources: Almasi and Gottlieb [1], Bertsekas and Tsitsiklis [3], Ortega [54], and Stone [66].

Parallel computers may be classified as *coarse grained* (a small number of very powerful processors joined together) and *massively parallel* (a very large number of inexpensive, and thus less powerful, processors). Parallel computers may also be divided into categories known as *Single Instruction Multiple Data* (SIMD) and *Multiple Instruction Multiple Data* (MIMD) (as opposed to Single Instruction Single Data [SISD] conventional uniprocessor computers). These names date to Flynn's article on multiprocessor machines [19]. As the names imply, all processors on SIMD machines execute a single instruction, albeit on different data, while the processors on MIMD machines may execute differ instructions. Computational tasks that fit naturally in an SIMD environment are known as *data parallel*; matrix multiplication and image processing (FFT) are two examples. At the present time, the Connection Machines made by the Thinking Machines Corporation, with up to 2^{16} processors in parallel, are the

most prominent SIMD machines.

While MIMD machines are more flexible and versatile, synchronization between processors is a major undertaking, especially as the number of processors is increased. Issues such as access to shared information by multiple processors and flow of information between processors must be resolved by the computer designer by implementing some protocol (the processors on an SIMD machine are synchronized in lockstep and the issues are quite different). As with roads, poor designs and heavy traffic can lead to flow control problems and gridlock. For an introduction to these issues, we direct the reader to [15].

Since vector dot products and matrix multiplications are data parallel, matrix construction is the major obstacle to the implementation of CDL-BIEM on SIMD machines. For simulation of a suspension of identical particles, the essential geometric information can be compressed and replicated in local memory; in fact, construction of the CDL-BIEM system matrix from the boundary element data is a data parallel operation. However, in the most general CDL-BIEM setting, the boundary element data may not fit into local memory. Another option would be to generate the matrix on another parallel computer and then perform parallel data input to the data vaults of an SIMD machine. On the other hand, with MIMD machines the time required for construction of each row is roughly the same, thus illustrating the point that details of the algorithm depend on the architecture. For yet another example, we mention that for computers with fast processors but limited memory, it may be more efficient to reconstruct the matrix at each iteration, while on most SIMD machines this tedious step should be avoided whenever possible. Finally, we can state with a high degree of confidence that with increasing problem and architecture complexity, communication between processors, that is, *flow*, rather than *processing* of information will be the bottleneck, so that reduction of information flow should be the goal in algorithm construction.

The basic premise that the simplest adaptations of CDL-BIEM are readily implemented on parallel machines has been confirmed, as was reported in Karrila *et al.* [39]. The computer used was a Sequent Symmetry, an MIMD machine consisting of 20 Intel 80386 processors, Weitek 1167 floating point accelerators and shared (common) memory. As the computational equivalent of about 60 MicroVAX IIs, the Symmetry is not the fastest computer for this problem. However, because of its low cost and rich menu of user-accessible parallel programming features, the Symmetry is a popular choice of computer scientists as a research tool for parallel processing studies.

The only essential modification required for this parallel machine was the introduction of *fork* and *barrier* statements, which split off and synchronize multiple processes. As the name implies, work on different processors stops at the barrier, thus allowing laggards to catch up with the others. This simplicity of code development is one measure of the parallel structure inherent in CDL-BIEM.

An efficiency based on actual speedup achieved divided by the number of processors used can be defined. In Table 19.3 we show the key result from

	Number of Processors		
N_s	1	8	16
1	649.68	83.89	43.55
2	1571.14	201.97	104.66
4	4146.77	530.66	273.78
8	12243.04	1559.13	797.17

Table 19.3: Benchmarks on the Sequent Symmetry: computation times (seconds) per iteration *vs.* number of spheres N_s, and number of processors.

[39]: computational times for multisphere mobilities as a function of number of (320-element) spheres and the number of processors used on the Sequent Symmetry. In all cases, efficiencies of about 96% were achieved. The iterative solutions converge so rapidly that system construction represents a significant fraction of the overall computational time, especially since matrix elements were recomputed at each iteration to conserve memory, as was the case for the results shown in Table 19.3. But since matrix construction and deflation are done in parallel, the overall algorithm remains quite efficient.

19.6 Reducing Communication Between Processors

19.6.1 Asynchronous Iterations

For small problems (say, for a cluster of several spheres), the matrix-vector multiplications of the Jacobi iterations are a straightforward operation. However when the method is directed at thousands of spheres, very large and dense systems result, say, on the level starting from 1,000,000 $\times$ 1,000,000. While it is exciting to contemplate that vector dot product computations of this magnitude can be done concurrently on parallel computers capable of executing these at the rate of 10^9 to 10^{12} floating point operations per second, we must address computational bottlenecks associated with communication limitations between processors.

Looking beyond the immediate problem at hand, we would like to address the critical problem of coordinating computational tasks (the matrix multiplication in the iterations) between hundreds and even thousands of powerful processors. As the processor components approach "supercomputer-on-a-chip" capabilities, it is the flow of information between these processors that becomes the weakest link of the *ab initio* simulation envisioned in this program.

In Figure 19.9, we show one way to minimize communications between processors. With n as the number of processors, the number of all possible connec-

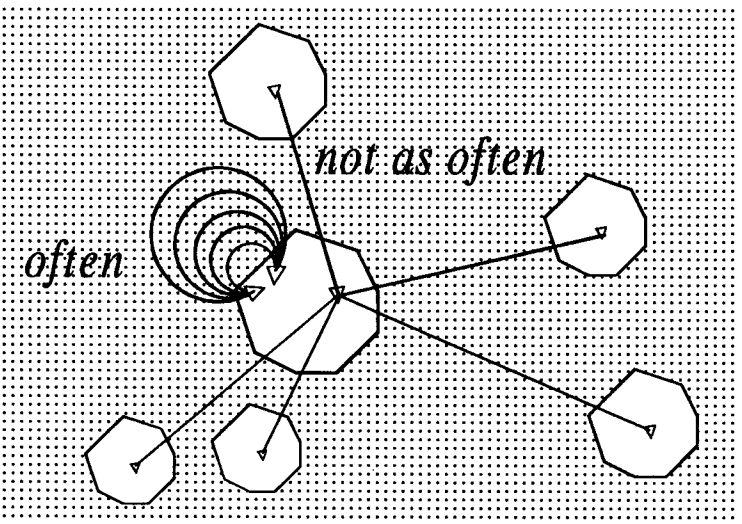

Figure 19.9: Asynchronous iterations to reduce communications and accelerate convergence.

tions grows as $n^2/2$, so that when thousands of processors are involved, communication becomes an important issue. In suspension simulation, the boundary elements on the same particle are strongly coupled, while those on different particles are less so, as we would expect on physical grounds. This implies that block Gauss–Seidel iterative strategies can be employed. For each particle, or cluster of neighboring particles, we allocate a processor with its own memory. Very rapid internal communications facilitate iterative improvements for each particle. Interactions between particles are obtained in a slower cycle, but the number of such cycles is greatly reduced. This approach raises two critical questions:

Acceleration. Do local iterations accelerate the rate of convergence? In calculating the rate of convergence, it seems natural to count only the "global iterations"; local iterations do not require communication between processors and are thus assigned zero cost.

Convergence. If information from distant parts of the suspension is not received and updated as often as local information, will the time delay cause non-convergence?

Fortunately, the answer to both of these is the favorable one: Local iterations greatly accelerate the rate of convergence; and despite the time delay from

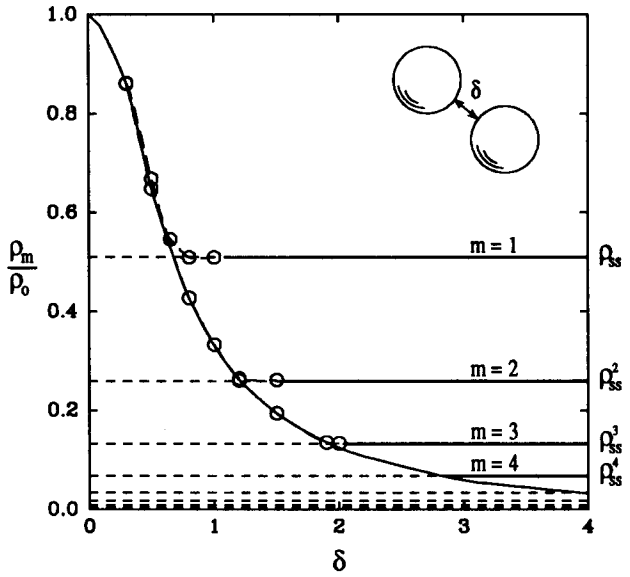

Figure 19.10: Collapsing the spectral radius with m local iterations.

distant regions, the iterations still converge to the correct solution. The key to both questions lies in the eigensystem of the double layer operator as perturbed by hydrodynamic interactions, which, for spheres, was addressed in Chapter 17. Since all combinations of multiparticle clusters involve a spectral radius less than one, the block Gauss–Seidel iterations will converge to a local fixed point. Furthermore, the action of the local iterations is now clear. Local iterations essentially map diagonal blocks to powers of the same block, and since these are the dominant factors behind the spectral radius, m local iterations should shrink the spectral radius ρ essentially as $\rho \rightarrow \rho^{m+1}$, as shown in Figure 19.10.

Indeed, the two-particle model problem suggests a fortuitous confluence of events. In the most natural parallel computational strategy envisioned for distributed processors, the time available for local iterations increases with particle-particle separation, so we can perform more local iterations. But with increasing particle-particle separations, there is also an enhanced benefit from such frequent local iterations. Indeed, when particles are greatly separated, with other particles in between, the ratio of local to global communication becomes so large that the time delay is essentially infinite; the interactions are screened. Since results obtained with and without screening can be compared, we have made a connection to one of the important concepts in effective medium theories. The ramifications for *ab initio* suspension simulation are significant.

19.6.2 Compactification of Distant Information

In the preceding discussion, we considered reducing the frequency of information exchange to address communication bottlenecks. Here, we consider reduction of the volume of information, *i.e.*, *compactification* of the information from distant regions, before sending them as message packets. Our idea bears strong resemblance to the multipole method employed by Greengard and Rokhlin to generate fast parallel codes for molecular dynamics simulations [26]. For suspended particles, the multipole expansions are a quite natural concept. Instead of the moment expansions of the intermolecular force laws for molecules in a cell volume, we do the same for the double layer density distribution over the surfaces of distant particles. The net effect is a great reduction in the amount of information required to characterize matrix elements linking all boundary elements on particles α with those on particle β. We simply store the coordinates of just the particle centers, and the first few multipole coefficients: force, torque, stresslet, quadrupoles, *etc.* But as always, these approximations to speed up the simulations can be introduced in a controlled fashion by performing tests with and without the approximation.

Exercises

Exercise 19.1 The Double Layer Density on a Small Polar Cap.
Consider the constant double layer density $\varphi = e_z$ distributed over the "polar cap" $0 \leq \theta \leq c \leq \pi$. Derive an analytical expression for $\mathcal{K}\varphi$ at the "north pole". Show that $\varphi + \mathcal{K}\varphi = 0$ when $c = \pi$ and that in the limit of small c, the result vanishes as $O(c^{3/2})$.

Exercise 19.2 The RBM Tractions of Platonic Solids.
Using the method illustrated in Chapter 16 involving the theorems of Lorentz and Riesz, determine numerically the RBM tractions of the platonic solids. Use mesh refinement to infer the nature of the surface tractions in the vicinity of the vertices.

Exercise 19.3 Rectangular Prisms.
Examine the numerical behavior of CDL-BIEM for rectangular prisms ranging from cubes to slender ribbons. Is it possible to draw general conclusions concerning the spectral radius of the deflated system as a function of the shape parameters of the prism?

References

[1] G. S. Almasi and A. Gottlieb. *Highly Parallel Computing.* Benjamin/Cummings, New York, 1989.

[2] K. E. Atkinson. *A Survey of Numerical Methods for the Solution of Fredholm Integral Equations of the Second Kind.* SIAM, Philadelphia, 1976.

[3] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation.* Prentice-Hall, Englewood Cliffs, NJ, 1989.

[4] R. B. Bird, W. E. Stewart, and E. N. Lightfoot. *Transport Phenomena.* Wiley, New York, 1960.

[5] E. Bodewig. *Matrix Calculus.* Interscience, New York, 1956.

[6] C. A. Brebbia and J. Dominguez. *Boundary Elements An Introductory Course.* McGraw-Hill, New York, 1989.

[7] H. Brenner. The Stokes resistance of an arbitrary particle — IV. Arbitrary fields of flow. *Chem. Eng. Sci.*, 19:703–727, 1964.

[8] L. J. Briggs and A. N. Lowan. *Tables of Associated Legendre Functions.* Columbia University Press, New York, 1945.

[9] P. C. Chan, R. J. Leu, and N. H. Zargar. On the solution for the rotational motion of an axisymmetric rigid body at low Reynolds number with application to a finite length cylinder. *Chem. Eng. Commun.*, 49:145–163, 1986.

[10] T. A. Cruse. Numerical solutions in three dimensional electrostatics. *Intl. J. Solids Structures*, 5:1259–1274, 1969.

[11] T. A. Cruse. An improved boundary-integral method for three dimensional elastic stress analysis. *Computers and Structures*, 4:741–754, 1974.

[12] T. Dabros. A singularity method for calculating hydrodynamic forces and particle velocities in low-Reynolds-number flows. *J. Fluid Mech.*, 156:1–21, 1985.

[13] L. M. Delves and J. Walsh. *Numerical Solution of Integral Equations.* Clarendon Press, Oxford, 1974.

[14] J. Dieudonne. *Foundations of Modern Analysis.* Academic Press, New York, 1960.

[15] A. Dinning. A survey of synchronization methods for parallel computers. *Computers*, 22:66–76, 1989.

[16] S. R. Dungan and H. Brenner. Force and torque on a body in an unbounded Stokes flow expressed explicitly in terms of respective quadratures of the pressure and vorticity fields. *Chem. Eng. Commun.*, 82:103–110, 1989.

[17] S. Fenyo and H. W. Stolle. *Theorie und Praxis der Linearen Integralgleichungen 4 (Theory and Practice of Linear Integral Equations 4).* Birkhauser, Basel, 1984.

[18] H. E. Fettis. A new method for computing toroidal harmonics. *Mathematics of Computation*, 24:667–670, 1970.

[19] M. J. Flynn. Very high-speed computers. *Proc. of the IEEE*, 54:1901–1909, 1966.

[20] Y. O. Fuentes. *Parallel Computational Strategies for Multiparticle Interactions in Stokes Flows.* Ph. D. Dissertation, University of Wisconsin, 1990.

[21] A. Friedman. *Foundations of Modern Analysis.* Dover reprint, New York, 1982.

[22] P. Ganatos, R. Pfeffer, and S. Weinbaum. A numerical-solution technique for three-dimensional Stokes flows, with application to the motion of strongly interacting spheres in a plane. *J. Fluid Mech.*, 84:79–111, 1978.

[23] F. R Gantmacher. *The Theory of Matrices.* Chelsea, New York, 1960.

[24] W. Gautschi. Computational aspects of three-term recurrence relations. *SIAM Review*, 9:24–82, 1967.

[25] I. S. Gradshteyn and I. M. Ryzhik. *Table of Integrals, Series, and Products.* Academic Press, New York, 1980.

[26] L. Greengard and V. Rokhlin. A fast algorithm for particle simulations. *J. Comp. Physics*, 73:325–348, 1987.

[27] N. M. Gunter. *Potential Theory.* Frederick Ungar Publishing, New York, 1967.

[28] P. S. Han and M. D. Olson. An adaptive boundary element method. *Int. J. for Num. Meth. in Eng.*, 24:1187–1202, 1987.

[29] J. Happel and H. Brenner. *Low Reynolds Number Hydrodynamics.* Martinus Nijhoff, The Hague, 1983.

[30] R. Hill and G. Power. Extremum principles for slow viscous flow and the approximate calculation of drag. *Q. J. Mech. and Appl. Math*, 9:313–319, 1956.

[31] A. S. Householder. *The Theory of Matrices in Numerical Analysis.* Dover, New York, 1975.

[32] R. Hsu and P. Ganatos. The motion of a rigid body in viscous fluid bounded by a plane wall. *J. Fluid Mech.*, 207:29–72, 1989.

[33] M. A. Jaswon and G. T. Symm. *Integral Equation Methods in Potential Theory and Elasticity.* Academic Press, New York, 1977.

[34] G. B. Jeffery. On the steady rotation of a solid of revolution in a viscous fluid. *Proc. London Math. Soc.*, 14(2):327–338, 1915.

[35] D. J. Jeffrey and Y. Onishi. Calculation of the resistance and mobility functions for two unequal rigid spheres in low-Reynolds-number flow. *J. Fluid Mech.*, 139:261–290, 1984.

[36] L. V. Kantorovich and G. P. Akilov. *Functional Analysis.* Pergamon Press, New York, 1982.

[37] S. J. Karrila. *Linear Operator Theory Applied to Fast Computational Strategies for Hydrodynamic Interactions in Viscous Flows.* Ph.D. Dissertation, University of Wisconsin, Madison, WI, 1988.

[38] S. J. Karrila and S. Kim. Integral equations of the second kind for Stokes flow: direct solution for physical variables and removal of inherent accuracy limitations. *Chem. Eng. Commun.*, 82:123–161, 1989.

[39] S. J. Karrila, Y. O. Fuentes, and S. Kim. Parallel computational strategies for hydrodynamic interactions between rigid particles of arbitrary shape in a viscous fluid. *J. Rheology*, 33:913–947, 1989.

[40] S. Kim. Singularity solutions for ellipsoids in low-Reynolds-number flows: with applications to the calculation of hydrodynamic interactions in suspensions of ellipsoids. *Intl. J. Multiphase Flow*, 12:469–491, 1986.

[41] S. Kim and R. T. Mifflin. The resistance and mobility functions of two equal spheres in low-Reynolds-number flow. *Phys. Fluids*, 28:2033–2045, 1985.

[42] S. Kim. Stokes flow past three spheres: an analytic solution. *Phys. Fluids*, 30(8):2309–2314, 1987.

[43] U. A. Ladyzhenskaya. *The Mathematical Theory of Viscous Incompressible Flow.* Gordon and Breach, New York, 1963.

[44] I. A. Lasso and P. D. Weidman. Stokes drag on hollow cylinders and conglomerates. *Phys. Fluids*, 29:3921–3934, 1986.

[45] H. A. Lorentz. Ein allgemeiner Satz, die Bewegung einer reibenden Flüssigkeit betreffend, nebst einigen Anwendungen desselben (A general theorem concerning the motion of a viscous fluid and a few applications from it). *Versl. Kon. Akad. Wetensch. Amsterdam*, 5:168–174, 1896. Also in *Abhandlungen über Theoretische Physik*, 1:23–42 (1907) and *Collected Papers*, 4:7–14, Martinus Nijhoff, The Hague, 1937.

[46] J. H. C. Luke. Convergence of a multiple reflection method for calculating Stokes flow in a suspension. *SIAM J. Appl. Math.*, 49:1635–1651, 1989.

[47] W. Magnus and F. Oberhettinger. *Formulas and Theorems for the Special Functions of Mathematical Physics.* Chelsea, New York, 1949.

[48] R. Mathon and R. L. Johnston. The approximate solution of elliptic boundary-value problems by fundamental solutions. *SIAM J. Numer. Anal.*, 14:638–650, 1977.

[49] M. Mayr. On the numerical solution of axisymmetric elasticity problems using an integral equation approach. *Mech. Res. Comm.*, 3:393–398, 1976.

[50] M. Mayr, W. Drexler, and G. Kuhn. A semianalytical boundary integral approach for axisymmetric elastic bodies with arbitrary boundary conditions. *Intl. J. Solids Structures*, 16:863–871, 1980.

[51] S. G. Mikhlin. *Linear Integral Equations.* Hindustan Publishing Corp., Delhi, 1960.

[52] S. G. Mikhlin. *Mathematical Physics, an Advanced Course.* North-Holland Publishing Co., New York, 1970.

[53] F. K. G. Odqvist. Über die Randwertaufgaben der Hydrodynamik zäher Flüssigkeiten (On the boundary value problems in hydrodynamics of viscous fluids). *Math. Z.*, 32:329–375, 1930.

[54] J. M. Ortega. *Introduction to Parallel and Vector Solution of Linear Systems.* Plenum Press, New York, 1988.

[55] I. G. Petrovskii. *Lectures on the Theory of Integral Equations.* Graylock Press, Rochester, NY, 1957.

[56] E. S. Pettyjohn and E. B. Christiansen. Effect of particle shape on free-settling rates of isometric particles. *Chem. Engr. Prog.*, 44:157–172, 1948.

[57] H. Power and G. Miranda. Second kind integral equation formulation of Stokes' flows past a particle of arbitrary shape. *SIAM J. Appl. Math*, 47:689–698, 1987.

[58] C. Pozrikidis. The instability of a moving viscous drop. *J. Fluid Mech.*, 210:1–21, 1990.

[59] J. M. Rallison. Note on the Faxén relations for a particle in Stokes flow. *J. Fluid Mech.*, 88:529–533, 1978.

[60] D. Ramkrishna and N. R. Amundson. *Linear Operator Methods in Chemical Engineering.* Prentice-Hall, Englewood Cliffs, NJ, 1985.

[61] F. J. Rizzo and D. J. Shippy. A boundary integral approach to potential and elasticity problems for axisymmetric bodies with arbitrary boundary conditions. *Mech. Res. Comm.*, 6:99–103, 1979.

[62] W. Rudin. *Functional Analysis.* Tata McGraw-Hill Publishing Comp, New Delhi, 1978.

[63] R. Schmitz and B. U. Felderhof. Friction matrix for two spherical particles with hydrodynamic interaction. *Physica*, 113A:103–116, 1982.

[64] D. J. Shippy, F. J. Rizzo, and R. K. Nigam. A boundary integral equation method for axisymmetric elastostatic bodies under arbitrary surface loads. In *Innovative Numerical Analysis for the Engineering Sciences (ed. R. Shaw, et al.)* University Press of Virginia, Charlotte, VA, 1980.

[65] C. Snow. *Hypergeometric and Legendre Functions with Applications to Integral Equations of Potential Theory.* NBS Appl. Math. Series 19, Washington DC, 1952.

[66] H. S. Stone. *High-Performance Computer Architecture.* Addison-Wesley, Reading, MA, 1987.

[67] A. H. Stroud. *Approximate Calculation of Multiple Integrals.* Prentice Hall, Englewood Cliffs, NJ, 1971.

[68] Symbolics Inc. *VAX UNIX MACSYMA Reference Manual: Version 11.* Symbolics, Inc., 1985.

[69] E. T. Whittaker and G. N. Watson. *A Course of Modern Analysis.* Cambridge University Press, New York, 1963.

[70] J. Wimp. *Computation with Recurrence Relations.* Pitman Advanced Publishing Progr, Boston, 1984.

[71] B. J. Yoon and S. Kim. Note on the direct calculation of mobility functions for two equal-sized spheres in Stokes flow. *J. Fluid Mech.*, 185:437–446, 1987.

[72] G. K. Youngren and A. Acrivos. Stokes flow past a particle of arbitrary shape: a numerical method of solution. *J. Fluid Mech.*, 69:377–403, 1975.