

ITERATIVE SOLUTION OF HERMITE BOUNDARY INTEGRAL EQUATIONS *

L. J. GRAY[†], S. NINTCHEU FATA[†], AND DING MA[‡]

Abstract. An efficient iterative method for the solution of the linear equations arising from a Hermite boundary integral approximation has been developed. Along with equations for the boundary unknowns, the Hermite system incorporates equations for the first order surface derivatives (gradient) of the potential, and is therefore substantially larger than the matrix for a corresponding linear approximation. However, by exploiting the structure of the Hermite matrix, a two-level iterative algorithm has been shown to provide a very efficient solution algorithm. In this approach, the boundary function unknowns are treated separately from the gradient, taking advantage of the sparsity and near-positive definiteness of the gradient equations. In test problems, the new algorithm significantly reduced computation time compared to iterative solution applied to the full matrix. This approach should prove to be even more effective for the larger systems encountered in three-dimensional analysis, and increased efficiency should come from pre-conditioning of the non-sparse matrix component.

Key words. boundary integral method, Hermite approximation, iterative solution.

AMS subject classifications. 65R20, 45E99

1. Introduction. The distinguishing feature of a cubic Hermite boundary integral approximation [16] is that the surface gradient of the primary function is obtained simultaneously with the solution of the boundary unknowns. For general boundary integral analysis, two motivations for pursuing this approximation scheme are to have a differentiable (C^1) interpolation of this primary function, *e.g.*, potential for the Laplace equation, displacement for elasticity, and to construct a higher order, more accurate, element using just the nodes employed for a simple linear element. Moreover, the Hermite approach is, for a number of reasons, attractive for moving boundary applications (see *e.g.* [5]): the gradient is generally required to compute surface velocities, and the higher accuracy and smoothness could be crucial for successfully evolving the boundary in time.

However, the price that is paid for the advantages of Hermite is rather steep: significantly increased computational costs associated with the construction, storage, and solution of a larger system of linear equations. For a two-dimensional scalar problem discretized with N boundary nodes, the resulting coefficient matrix is of order $3N$, compared to N for a corresponding linear approximation. For a vector problem these numbers are $6N$ versus $2N$, and in three dimensions the matrix order increases by a factor of 4 over a linear analysis.

Nevertheless, from the work in [8] (this reference also contains a more complete discussion of previous work on Hermite methods), the penalties associated with constructing and storing the extra boundary integral gradient equations have largely been removed. The gradient equations can now be expressed solely in terms of ‘local’ singular integrals, avoiding the usual complete boundary integration and moreover producing sparse, rather than dense, set of linear equations [10]. The

* This work was supported by the Office of Advanced Scientific Computing Research, U.S. Department of Energy, under contract DE-AC05-00OR22725 with UT-Battelle, LLC.

[†]Computer Science and Mathematics Division, Oak Ridge National Laboratory, Oak Ridge, TN 37831-6367
ljg@ornl.gov

[‡]Farragut High School, Farragut TN.

computation time to assemble these equations is therefore $\mathcal{O}(N)$ rather than $\mathcal{O}(N^2)$, and if stored in sparse format, the storage cost is also $\mathcal{O}(N)$.

This paper now addresses the remaining drawback of Hermite, the computational cost in solving the larger system of linear equations. As the gradient equations in [10] are sparse, this suggests, if not demands, an iterative solution of the linear system. Moreover, as will be discussed further below, the principal sub-matrices for the gradient values are well-conditioned and the equations for the separate components of the gradient are only weakly coupled. While an iterative solution of the full Hermite matrix can take advantage of the sparsity, it would not exploit these other features of the system. A ‘two-level’ iterative scheme is therefore proposed, the basic idea being to separate the solution of the gradient equations from the boundary integral equation. Tests employing the two-dimensional Laplace equation will compare the performance of this algorithm with the application of the iterative solver to the full Hermite matrix.

The paper is organized as follows. The next section very briefly summarizes the Hermite algorithm presented in [8], the reader is asked to consult this reference for further details. Section 3 describes the new iterative algorithm in the context of the two-dimensional Laplace equation. However, there is nothing specific to this equation, and the methods also extend directly to three dimensions. Test results, for three-dimensional axisymmetric geometries as well as two-dimensional, are discussed in Section 4, to be followed by some concluding remarks.

2. Hermite Interpolation. For the Laplace equation $\nabla^2\phi = 0$, $\phi = \phi(x, y)$, the *exterior limit* boundary integral equation can be written as [7]

$$\lim_{P_E \rightarrow P} \int_{\Gamma} \phi(Q) \frac{\partial G}{\partial \mathbf{n}}(P_E, Q) dQ - \int_{\Gamma} G(P, Q) \frac{\partial \phi}{\partial \mathbf{n}}(Q) dQ = 0, \quad (2.1)$$

where the Green’s function is

$$G(P, Q) = -\frac{1}{2\pi} \log(\|Q - P\|) = -\frac{1}{2\pi} \log(r), \quad (2.2)$$

and P_E are *exterior* points converging to P . Following standard practice [3], a finite system of linear equations is obtained by approximating the boundary and the boundary functions. Herein, as in [8], a linear interpolation is employed for the surface flux, while cubic Hermite shape functions $\psi_j(t)$, $0 < t < 1$,

$$\begin{aligned} \psi_1(t) &= (1 + 2t)(1 - t) \\ \psi_2(t) &= t^2(3 - 2t) \\ \psi_3(t) &= t(1 - t)^2 \\ \psi_4(t) &= -t^2(1 - t). \end{aligned} \quad (2.3)$$

are employed to define the approximate boundary

$$Q(t) = (x(t), y(t)) = \sum_{j=1}^2 (x_j, y_j) \psi_j(t) + \sum_{j=3}^4 (a_j, b_j) \psi_j(t) \quad (2.4)$$

and the interpolation of the potential function

$$\phi(Q(t)) = \sum_{j=1}^2 \phi(Q_j) \psi_j(Q) + \sum_{j=3}^4 \frac{d\phi}{dt}(Q_{j-2}) \psi_j(Q). \quad (2.5)$$

In the above, the two nodes defining the element are $Q_1 = (x_1, y_1)$ and $Q_2 = (x_2, y_2)$ and the coefficients (a_3, b_3) and (a_4, b_4) are defined by specifying the unit normals at the nodes. In addition, the tangential derivatives in Eq. (2.5) are given in terms of the nodal gradients,

$$\frac{d}{dt}\phi(t) = \frac{d}{dt}\phi(x(t), y(t)) = \frac{\partial\phi}{\partial x}x'(t) + \frac{\partial\phi}{\partial y}y'(t), \quad (2.6)$$

and thus from Eq. (2.4)

$$\begin{aligned} \frac{d\phi}{dt}(Q_1) &= a_3 \frac{\partial\phi}{\partial x}(Q_1) + b_3 \frac{\partial\phi}{\partial y}(Q_1) \\ \frac{d\phi}{dt}(Q_2) &= a_4 \frac{\partial\phi}{\partial x}(Q_2) + b_4 \frac{\partial\phi}{\partial y}(Q_2). \end{aligned} \quad (2.7)$$

Finally, Eq. (2.1) and the gradient equation discussed below are approximated using a Galerkin approximation [3, 7].

2.1. Gradient equations. As discussed in detail in [10], the gradient equations exploit the boundary limit formulation of the integral equations. Specifically, the difference of interior and exterior limits results in

$$\nabla\phi(P) + \left[\lim_{P_I \rightarrow P} - \lim_{P_E \rightarrow P} \right] \int_{\Gamma} \left\{ \phi(Q) \nabla \frac{\partial G}{\partial \mathbf{n}}(P, Q) - \nabla G(P, Q) \frac{\partial \phi}{\partial \mathbf{n}}(Q) \right\} dQ = 0. \quad (2.8)$$

Note that the only integrations that survive the difference of the limits are singular, all nonsingular integrations (and many singular integrals as well) are the same for either limit and must vanish. The resulting matrix rows are therefore sparse. The second aspect of this equation that will prove useful is that the matrix elements arising from the free term $\nabla\phi(P)$ outside the integral (and therefore associated with boundary gradient values) are the dominant contributions to the gradient equations. Moreover, as the approximation of the surface gradient is linear, these matrix elements are simply integrals of the form

$$\int \psi_k(P) \psi_j(P) dP, \quad (2.9)$$

where $\psi_k(P)$ are the linear element shape functions. This dominant free term (by itself) therefore leads to a symmetric positive definite matrix.

The linear equations resulting from the Hermite algorithm therefore consist of three (in two dimensions) parts, the boundary integrals for the unknown values of potential or flux, together with derivative equations for the x and y components of the gradient. This structure will be exploited in the algorithm discussed in the next section.

3. Iterative Algorithm. If the boundary is discretized with N nodes, the Hermite algorithm results in a $3N$ by $3N$ system of linear equations $Ax = b$. It is convenient to write these equations in the form

$$\begin{pmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}, \quad (3.1)$$

where each A_{ij} submatrix is $N \times N$. The vector x_1 represents the unknown boundary values of potential or flux, and the first block row of equations is the discretized form of Eq. (2.1). Similarly, x_2 and x_3 are vectors containing the x - and y -components of the gradient, and the second and third rows are obtained from Eq. (2.8).

As noted above, the gradient equations only involve local singular integrals, and thus all A_{2j} and A_{3j} are sparse matrices. Moreover, if a linear interpolation had been employed to approximate the potential in the gradient equations, then the only matrix elements multiplying the gradient values would be from the free term in Eq. (2.8). Consequently, the diagonal blocks A_{22} and A_{33} would be symmetric positive definite and moreover $A_{23} = A_{32} = 0$. As it is expected that the Hermite sub-matrices are ‘small’ perturbations of the corresponding linear ones, the x - and y -component gradient equations should be only weakly coupled, and A_{22} , A_{33} should be well-conditioned.

The proposed algorithm therefore assumes an initial guess for x_2 and x_3 and then solves for x_1 from

$$A_{11}x_1 = b_1 - A_{12}x_2 - A_{13}x_3 . \quad (3.2)$$

The values of x_2 and x_3 can then be updated by solving

$$\begin{aligned} A_{22}x_2 &= b_2 - A_{21}x_1 - A_{23}x_3 \\ A_{33}x_3 &= b_3 - A_{31}x_1 - A_{32}x_2 \end{aligned} \quad (3.3)$$

and the whole process (termed an *outer iteration*) repeated until convergence. As just noted, the two systems in Eq. (3.3) should be well-conditioned, and the coupling between them, provided by A_{23} and A_{32} , very weak. Thus, there is reason to expect that this ‘two-level’ iteration will first of all converge, and second, be efficient.

Regarding computational cost, note that if all sparse matrix-vector multiplications are ignored, applying an iterative solver to the full Hermite matrix requires $3N^2$ operations for each matrix-vector multiplication. On the other hand, Eq. (3.2) requires only N^2 operations per iteration, plus $2N^2$ operations to compute the right hand side. Thus, depending upon the convergence, the two-level scheme could provide considerable savings.

3.1. Stopping Criteria. The iterative solver employed for the new algorithm, Eq. (3.2) and Eq. (3.3), and for comparison purposes the full Hermite matrix, is (Sylvain to add stuff here).

For the BiStab algorithm applied to $Ax = b$, the iteration terminates when the Euclidean norm of the residual vector r_k satisfies

$$\|r_k\| \leq \epsilon \|b\| , \quad (3.4)$$

and ϵ is a supplied tolerance.

For the two-level algorithm, we apply the same residual criterion as above. Note that this requires an additional $3N^2$ operations to compute the global residual for every outer iteration of this algorithm. However, the test cases indicate that relatively few outer iterations are necessary, and thus this expense will not be critical.

The residual tolerances prescribed for the ‘inner’ solutions of Eq. (3.2) and Eq. (3.3) need not be the same as the global tolerance, *i.e.* they can be adjusted within the outer iteration. Early on the

right hand side values will not be correct, and thus clearly there is no point in converging with very high accuracy. The value of ϵ employed for these initial solutions can therefore start off relatively large and then decrease each outer iteration. This extra leeway in the algorithm will be exploited in the results presented below.

4. Test Calculations.

4.1. Two-dimensional. The first set of test calculations employs the two-dimensional Hermite algorithm described in [8] for the Laplace equation. In addition, the initial calculations have Dirichlet boundary conditions posed on the unit disk $x^2 + y^2 = 1$ and the ellipse $x^2 + 4y^2 = 1$, the known boundary potential being $\phi = x^2 - y^2$. Uniform meshes with N nodes are employed for the disk, whereas the elements for the ellipse were defined in terms of a constant central, and are therefore non-uniform. The two-level algorithm will be compared with a direct application of the iterative solver to the full Hermite matrix, and in both cases the tolerance for the residual was $\epsilon = 10^{-10}$. However, as noted above, it is not necessary or desirable to solve Eq. (3.2) and Eq. (3.3) to this level of accuracy throughout; in these tests we have set the tolerance for the inner solutions to be $\epsilon = 10^{-m}$, where m is either the outer iteration number or 10, whichever is smaller.

TABLE 4.1

Computational cost (in units of N^2) and residuals for iterative solution methods, $\epsilon = 10^{-10}$. The domain is the unit disk.

N	Full Matrix		Two-Level	
	C_I	Residual	C_I	Residual
100	219	6.96 (-11)	60	6.31 (-12)
200	483	4.67 (-11)	75	6.52 (-12)
300	387	3.62 (-11)	84	2.37 (-11)
400	555	3.98 (-11)	100	5.58 (-12)
500	459	3.93 (-11)	100	1.22 (-12)
600	555	3.34 (-11)	107	7.37 (-12)
800	771	3.16 (-11)	93	1.60 (-11)

Tables 4.1 and 4.2 list the computational work C_I – again, ignoring all sparse matrix operations – and the residuals for solving the Dirichlet problem. The costs are given in terms of N^2 , and thus C_I for the full matrix method is three times the number of iterations required to obtain a converged solution. For the two-level scheme, C_I consists of the total number of matrix-vector multiplies involving A_{11} , plus five times the number of outer iterations. This contribution is due to the $2N^2$ needed per outer iteration to re-compute the right hand side in Eq. (3.2), plus $3N^2$ to compute the global residual.

With a few exceptions, either iterative scheme involves much less work than the $(3N)^3/3$ required by direct factorization. However, the ‘divide and conquer’ approach of the two-level algorithm is clearly significantly faster than working with the entire matrix. The success of this approach stems from the fact that, at least in these tests, very few outer iterations are required to reach convergence. Table 4.3 lists the number of outer iterations required for the problem on the unit disk, along with the total number of matrix-vector multiplications required to solve the two systems in Eq. (3.3).

The very small number of outer iterations is clearly key to the efficiency of the algorithm, and this is

TABLE 4.2

Computational cost (in units of N^2) and residuals for iterative solution methods, $\epsilon = 10^{-10}$. The domain is an ellipse.

N	Full Matrix		Two-Level	
	C_I	Residual	C_I	Residual
100	1100	3.33 (-11)	163	6.31 (-11)
150	990	4.45 (-11)	118	2.19 (-11)
200	894	2.38 (-11)	134	4.15 (-12)
300	1100	2.64 (-11)	174	1.97 (-12)
600	1518	4.48 (-11)	82	5.38 (-11)

TABLE 4.3

Outer iterations and iterations required to solve the gradient equations for the problem posed on the unit disk, $\epsilon = 10^{-10}$.

N	Outer	A_{22}	A_{33}
100	3	27	51
200	3	27	43
300	3	27	43
400	3	27	43
500	3	27	35
600	3	27	35
800	3	27	51

likely due to weak coupling with the gradient equations: initially solving A_{11} by itself must get very close to the correct x_1 . What is also noteworthy about these numbers is that they are, compared to the iterations required to solve Eq. (3.2), largely independent of the mesh. In fact, for the ellipse tests these numbers are constants, the values corresponding to a row in Table 4.3 being 3, 27 and 43. The good conditioning of A_{22} and A_{33} certainly plays a role here, and again justifies the splitting of the A matrix. Moreover, it also indicates that the challenge of further expediting the linear algebra by developing a good pre-conditioner (note that no pre-conditioning has been employed) can focus solely on the A_{11} matrix.

The two iterative procedures have also been applied to a Neumann problem posed on the unit disk. The differences compared to the Dirichlet problem are in the first block row A_{1k} and column A_{k1} matrices, the integrals that comprise these matrices now originating from the integrals of the Green's function in Eq. (2.1), rather than its normal derivative. The results are shown in Table 4.4, and again indicate that the two-level algorithm is quite effective.

4.2. Three-dimensional Axisymmetric. The new iterative algorithm was also tested using the Hermite approximation for three-dimensional axisymmetric problems [9]; this reference discusses a linear element implementation, but the extension to Hermite is relatively straightforward. Although these calculations are still essentially two-dimensional, they involve different Green's function kernels and, moreover, these kernels have different behavior, depending upon whether the source is near the symmetry axis or not [1, 6, 2].

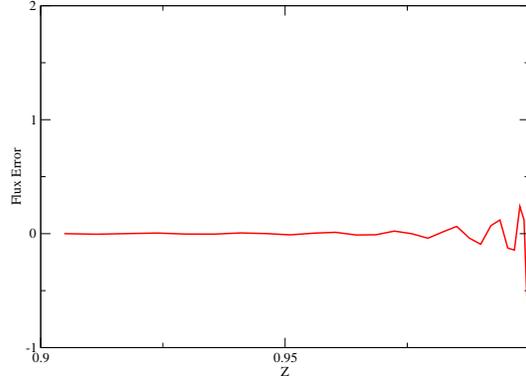


FIG. 4.1. Error in the computed flux (for the last iterate) near the symmetry axis for the axisymmetric sphere problem.

TABLE 4.4

Computational cost (in units of N^2) and residuals for iterative solution methods, $\epsilon = 10^{-10}$. The problem is a circle with Neumann boundary data.

N	Full Matrix		Two-Level	
	C_I	Residual	C_I	Residual
100	75	3.74 (-13)	42	3.73 (-11)
200	51	6.88 (-11)	42	2.94 (-12)
300	75	1.09 (-11)	42	6.51 (-12)
400	75	4.59 (-14)	28	6.09 (-11)
500	75	3.22 (-12)	28	2.42 (-11)
600	51	2.16 (-11)	28	5.77 (-11)
800	75	3.87 (-12)	28	1.43 (-11)

TABLE 4.5

Computational cost (in units of N^2) and \mathcal{L}_2 errors for the new iterative algorithm, $\epsilon = 5 \times 10^{-7}$, the Full Matrix failed to converge. The problem is a sphere with Dirichlet boundary data.

N	C_I	\mathcal{L}_2 Error		
		ϕ_n	ϕ_r	ϕ_z
100	813	7.22 (-4)	2.18 (-4)	2.51 (-4)
200	1514	2.04 (-4)	4.46 (-4)	1.48 (-4)
400	1220	1.27 (-4)	1.90 (-4)	1.17 (-4)
600	1213	2.41 (-4)	2.92 (-4)	2.40 (-4)

The first test is for a sphere, with Dirichlet boundary conditions $\phi = x^2 + y^2 - 2z^2$, or in axisymmetric notation, $\phi = r^2 - 2z^2$. The two-dimensional boundary is therefore the arc $r^2 + z^2 = 1$, $r \geq 0$. Quite surprisingly for this very simple geometry, the straightforward application of the iterative solver, with

the residual tolerance set at 5×10^{-7} , failed to converge. The problem appears to be connected to the symmetry axis: Fig. 4.1 plots the solution vector (flux and gradient) obtained after the maximum allowed 5000 iterations, and it appears reasonable except near the endpoints of the domain, located at the symmetry axis. Presumably the trial solutions produced by the iterative solver have converged everywhere else but fluctuate near the axis. It is quite possible that this behavior is a consequence of having chosen the Galerkin weight function to be zero at the axis, see [9].

TABLE 4.6

Computational cost (in units of N^2) and \mathcal{L}_2 errors for the two algorithms, $\epsilon = 5 \times 10^{-7}$. The problem is a torus with Dirichlet boundary data.

N	C_I		\mathcal{L}_2 Error		
	Full Matrix	Two Level	ϕ_n	ϕ_r	ϕ_z
100	246	79	2.94 (-3)	4.65 (-4)	9.29 (-4)
200	198	42	7.34 (-4)	1.16 (-4)	2.32 (-4)
600	198	42	8.13 (-5)	1.35 (-5)	2.59 (-5)

By contrast, the two-level algorithm does manage to converge. However, as indicated by the large number of iterations and the fluctuations in the \mathcal{L}_2 errors, Table 4.5, this algorithm is struggling. An inspection of the iterates again indicated that the difficulties are confined to the region near the symmetry axis, and thus both algorithms were run for the same Dirichlet problem on a torus (inner radius 1 and outer radius 3); the boundary is therefore a circle having no contact with the symmetry axis. The results are shown in Table 4.6. Both algorithms now perform very well, and as with previous tests, the new method requires significantly less work. The \mathcal{L}_2 errors for the two methods were nearly identical, and thus only those for the two level algorithm are shown.

5. Conclusions. The sparse gradient equations developed previously in [10] reduced to a reasonable level the prohibitive cost of constructing and storing the Hermite system matrix. It has now been shown that the time required to solve this system can be reduced by exploiting the structure of this system. Based upon tests for the two-dimensional and three-dimensional axisymmetric Laplace equation, the two-level iterative algorithm described herein appears to be significantly more efficient than a iterative solution of the complete Hermite system.

It is expected that this new algorithm will permit, and be even more effective for, the application of the Hermite approximation in other situations. In three dimensions, the Hermite system can obviously become quite large, especially for a vector problem. In this case there are three component functions, each requiring its own gradient calculation.

The Hermite system can also become much larger in another manner. Note that as in other implementations [4, 13, 14], we have only employed the Hermite approximation to the boundary potential; a linear interpolation has been employed for the flux. The reason is that applying Hermite to the normal derivative function requires the ability to compute second order derivatives of the potential. However, it has been recently shown that the techniques employed for the gradient, most importantly the generation of sparse equations, can be extended to these higher order derivatives [12]. Constructing a Hermite interpolation for the flux therefore becomes a possibility, but the complete matrix system will be huge: in two dimensions there are three second order derivatives, and six in three dimensions. The new algorithm will hopefully reduce the associated linear algebra costs to the point that this complete Hermite algorithm would be feasible.

Finally, note that this formulation and solution of the Hermite system is compatible with fast methods for the solution of the boundary integral equation, Eq. (2.1). With a fast method, *e.g.* Fast Multipole [11] or FFT approaches [15], the first row A_{1k} is not fully assembled, and the required matrix vector multiplications can be carried out with $\mathcal{O}(N)$ operations rather than N^2 . In this case, the number of iterations is then roughly a rough measure of the computational cost, and the two-level algorithm still appears to perform better.

Acknowledgment. This research was supported by the Office of Advanced Scientific Computing Research, U.S. Department of Energy, under contract DE-AC05-00OR22725 with UT-Battelle, LLC. The participation of the third author was through an appointment to the Higher Education Research Experiences Program (HERE) at Oak Ridge National Laboratory.

REFERENCES

- [1] A. A. BAKR, *The boundary integral equation method in axisymmetric stress analysis*, Springer-Verlag, Berlin, 1986.
- [2] J. BALÁŠ, J. SLÁDEK, AND V. SLÁDEK, *Stress analysis by the boundary element method*, Elsevier, Amsterdam, 1989.
- [3] M. BONNET, *Boundary Integral Equation Methods for Solids and Fluids*, Wiley and Sons, England, 1995.
- [4] J. F. DURODOLA AND R. T. FENNER, *Hermitian cubic boundary elements for two dimensional potential problems*, Int. J. Numer. Meth. Engrg., 30 (1990), pp. 1051–1062.
- [5] M. GARZON, D. ADALSTEINSSON, L. J. GRAY, AND J. A. SETHIAN, *A coupled level-set boundary integral method for moving boundary simulations*, Interfaces and Free Boundaries, 7 (2005), pp. 277–302.
- [6] E. GRACIANI, V. MANTIČ, F. PARIS, AND A. BLÁQUEZ, *Weak formulation of axi-symmetric frictionless contact problems with boundary elements. Application to interface cracks*, Computers and Structures, 83 (2005), pp. 836–855.
- [7] L. J. GRAY, *Evaluation of singular and hypersingular Galerkin boundary integrals: direct limits and symbolic computation*, in Singular Integrals in the Boundary Element Method, V. Sladek and J. Sladek, eds., Advances in Boundary Elements, Computational Mechanics Publishers, 1998, ch. 2, pp. 33–84.
- [8] L. J. GRAY AND M. GARZON, *On a Hermite boundary integral approximation*, Computers and Structures, 83 (2005), pp. 889–894.
- [9] L. J. GRAY, M. GARZON, V. MANTIČ, AND E. GRACIANI, *Galerkin boundary integral analysis for the axisymmetric Laplace equation*, Int. J. Numer. Meth. Engrg., 66 (2006), pp. 2014–2034.
- [10] L. J. GRAY, A.-V. PHAN, AND T. KAPLAN, *Boundary integral evaluation of surface derivatives*, SIAM J. Sci. Comput., (2005). in press.
- [11] L. GREENGARD AND V. ROKLIN, *A fast algorithm for particle simulations*, J. Comp. Phys., 73 (1987), pp. 325–348.
- [12] M. N. J. MOORE, L. J. GRAY, AND T. KAPLAN, *Evaluation of supersingular integrals: second order boundary derivatives*, Int. J. Numer. Meth. Engrg., xx (2006), pp. xxxx–xxxx.
- [13] K. H. MUCI-KÜCHLER AND T. J. RUDOLPHI, *Coincident collocation of displacement and tangent derivative boundary integral equations in elasticity*, Int. J. Numer. Meth. Engrg., 36 (1993), pp. 2837–2849.
- [14] ———, *Application of tangent derivative boundary integral equations to the formulation of higher order boundary elements*, Int. J. Solid Struct., 31 (1994), pp. 1565–1584.
- [15] J. R. PHILLIPS AND J. K. WHITE, *A precorrected-FFT method for electrostatic analysis of complicated 3-D structures*, IEEE Trans. Comput. Aided Design of Integrated Circuits and Systems, 16 (1997), pp. 1059–1071.
- [16] J. O. WATSON, *Hermitian cubic and singular elements for plain strain*, in Developments in Boundary Element Methods - Vol. 4, P. K. Banerjee and J. O. Watson, eds., Elsevier Applied Science Publishers, London and New York, 1986, ch. 1, pp. 1–28.