On the use of multipole methods for domain integration in the BEM

A.A. Mammoli, M.S. Ingber & M.J. Brown
Department of Mechanical Engineering, University of New Mexico, USA

Abstract

Often, a single application of the reciprocity theorem results in an integral equation that contains a domain integral, which must be evaluated in order to obtain the boundary solution. Because of aesthetic reasons, dual or multiple reciprocity has been used to remove the domain integral, thus restoring the ‘boundary only’ formulation. However, this improvement in elegance is accompanied by certain limitations in terms of the size and type of problem that can be treated, and in the accuracy of the solution.

Here, traditional domain mesh based integration, in combination with a Barnes-Hut [1] multipole scheme that has the purpose of reducing the operation count, is used in order to evaluate the domain integral. It is shown that this method is both considerably faster and more accurate than meshless methods.

The solution of nonlinear problems can speeded up further by avoiding collocation in the interior, and evaluating the interior field by successive substitution. This method also has increased flexibility in terms of equations that can be treated. However, it is found that convergence is not always achieved.

1 Introduction

The boundary element method (BEM) was originally developed to treat homogeneous linear partial differential equations for which a fundamental solution associated with the adjoint operator could either be determined or approximated. Nonlinear and nonhomogeneous problems can also be treated by supplementing the boundary integral equation with a domain integral. The standard evaluation of the domain integral has generally been seen as going against the spirit of the
BEM. As a result, several methods for eliminating domain integrals associated with boundary element methods have been developed over the years. Nardini and Brebbia [8] proposed a generalization of the concept of particular integrals, which they interpreted as a localized particular solution approach in order to eliminate the domain integral in a free vibration problem. The method, which they called the dual reciprocity method (DRM), was extended to parabolic problems by Wrobel and Brebbia [10, 11]. Partridge, Brebbia, and Wrobel [9] wrote a book on the DRM detailing the use of the method for a wide variety of problems. The DRM has probably been the most popular of the methods to deal with domain integrals in boundary element methods as evidenced by the sheer mass of journal and conference papers written about the method.

Recently, some developments have occurred which may revive the interest in domain integration. First, new meshing techniques allow faster and more efficient domain meshing to be performed, even on very complex geometries. Secondly, the application of fast multipole methods to the evaluation of domain integrals can bring about improvements in calculation speed of orders of magnitude, thereby making domain integration an attractive alternative to meshless methods. Fast multipole methods have recently been applied to evaluate boundary integral equations. Applications of fast multipole methods for boundary element analysis have been presented, among others, by Korsmeyer [6] to study 3D potential problems, Gómez and Power [3] for Stokes flow problems, and Mammoli and Ingber [7] for suspension problems. In all of these applications, the use of the multipole methods were limited strictly to boundary integrals and not to any associated domain integrals.

In the current paper, four methods for evaluating domain integrals occurring in boundary element methods are compared. These methods include domain integration using Gaussian quadrature, with and without multipole acceleration, the dual reciprocity method, and the particular solution method. The methods are compared in terms of ease of use, accuracy, robustness, and CPU requirements. Only the implementation of the multipole-accelerated classical domain integration scheme is presented here, as the implementation of the dual reciprocity and particular solution methods is available elsewhere [4].

2 Numerical formulations

The classical domain integration approach for a nonlinear problem requires the direct evaluation of a domain integral, where the integrand is the product of a function \( f(z) \), where \( z \) is some known or unknown interior field, depending on the problem being treated, and a kernel function. The integral can be evaluated by quadrature, and by regularization of weakly singular integrals using coordinate transformation.

In terms of the discretized system, the domain integral can be viewed as simply adding to the load vector. The system of equations that results from the discretization of a nonlinear boundary integral equation (BIE) applied to all boundary
collocation points is of the form:

\[ Ax = b - f(z), \]

where \( A \) is a coefficient matrix, \( b \) is the vector that results from known boundary conditions, and \( f(z) \) is a vector that results from \( z \).

In the case of a known integrand, the solution of the system is trivial. In the more interesting case that the integrand in the domain integral is not known \textit{a priori}, one can adopt two strategies:

1. Direct collocation: if \( f(z) \) is linear, the values of \( z \) at the interior finite element modes are easily introduced as additional unknowns. A linear system of equations are generated by collocating the BIE at both the boundary element nodes and the interior finite element nodes:

\[
\begin{bmatrix}
A & C \\
B & D
\end{bmatrix}
\begin{bmatrix}
x \\
z
\end{bmatrix} = 
\begin{bmatrix}
b \\
d
\end{bmatrix},
\]

where the matrices \( A \) and \( C \) result from collocation points on the boundary and integration on the boundary and in the interior respectively, while the matrices \( B \) and \( D \) result from collocation points in the interior and integration on the boundary and in the interior respectively. The solution of the system yields the boundary solution and the values of \( z \) simultaneously.

2. Iteration by successive substitution: In some cases, a solution can be obtained iteratively by evaluating \( f(z) \) using an estimate of \( z \). An updated estimate of \( z \) can then be obtained by postprocessing the boundary solution, and substituted again into the boundary problem. This method has been used successfully for various problems, for example by Bush and Tanner [2].

Independently of which method is chosen, the evaluation of the domain integral itself presents two difficulties: interior discretization and computational cost. The difficulty of the discretization is a function of the complexity of the geometry. Meshing algorithms have progressed considerably in recent years, and most geometries can be meshed efficiently using unstructured meshes.

In this paper, the latter concern is addressed. The evaluation of the domain integral can be speeded up considerably by using the Barnes–Hut approximation. The operation count for the evaluation of the domain integral per collocation node is reduced from \( O(M) \) to \( O(\log M) + N \), where \( M \) is the number of interior nodes and \( N \) is the number of boundary nodes. A consequence of the multipole evaluation of the domain integral is that the matrices that result from the integration over the interior domain (\( C \) and \( D \)) are not stored explicitly, precluding a direct evaluation of the solution of the linear system. In fact one is limited to using solvers where only the forward matrix–vector product is required. Because of its robustness, a generalized minimal residual (GMRES) algorithm is used here.

The details of the Barnes-Hut multipole expansion have been explained in detail elsewhere [3, 7], and the application of the multipole methods to domain integrals is straightforward. However, a short explanation is given in the interest of
The integral over a domain element $\Omega_l$, shown in Fig. 1, can be approximated by:

$$
\int_{\Omega_l} u^*(\xi, \eta) f(\eta, z) d\Omega \approx \sum_{q=0}^{n} \frac{\partial^q}{\partial \xi_{k_1} \partial \xi_{k_2} \ldots \partial \xi_{k_q}} u^*(\xi, \eta_0) C^l_{k_1 k_2 \ldots k_q},
$$

where the coefficients $C_{k_1 k_2 \ldots k_q}$ are moment tensors given by:

$$
C^l_{k_1 k_2 \ldots k_q} = \int_{\Omega_l} \tau_{k_1} \tau_{k_2} \ldots \tau_{k_q} f(\eta, z) d\Omega.
$$

The variation of $f(\eta, z)$ is accounted for by appropriate interpolation functions, quadratic isoparametric in this case.

Moment tensors about a point $\eta_1$ can be calculated economically by modifying the moment tensors about point $\eta_0$, without the need to re-evaluate the integrals. By 'shifting' the moment tensors from the local point $\eta_0$ for each subdomain to a common point $\eta_1$, the integral from a number of subdomains can now be evaluated with a single tensor/derivative multiplication, as shown in Fig. 1. The same procedure can be used repeatedly, allowing the evaluation of integrals over large portions of the entire domain in a single set of tensor/derivative products. The size of such clusters of elements is limited by the size of the truncation error $E_n$, which is a function of the separation of the cluster from the collocation point $\xi$. Far away from $\xi$, larger clusters can be constructed.

The domain integral is now evaluated in two parts, a near field and a far field. In the near field, the classical domain integration is performed. In the far field, multipole expansions are used. The integration strategy can be summarized by:

$$
\int_{\Omega} u^*(\xi, \eta) h(\eta, u) d\Gamma_y = \sum_{i \in n} \int_{\Omega} u^*(\xi, \eta) h(\eta, u) d\Gamma + \sum_{j \in f} \sum_{q=0}^{n} C^j_{k_1 k_2 \ldots k_q} \frac{\partial^q}{\partial \xi_{k_1} \partial \xi_{k_2} \ldots \partial \xi_{k_q}} u^*(\xi, \eta_j)(5)
$$
Each collocation node is associated with a list of near-field elements (nf) and a list of far-field (ff) clusters. The number of near-field elements in the list remains approximately constant as the problem size increases, while the number of far-field clusters grows as $\log M$, resulting in an $O(\log M)$ scaling overall. Care must be taken in order to make the generation and storage of near-field and far-field element and cluster lists economical. The advantage of the multipole expansion can be severely curtailed if lists are generated and stored inefficiently. An example of a near-field element list and a far-field cluster list is shown in Fig. 2.

Figure 2: Clustering for the domain integral for the node highlighted by the dot. Notice that the number of clusters (light grey) is far smaller than the number of far-field elements that they replace (dark grey). Matrix entries are stored only for the near-field elements (white).

3 Test problem and results

The test problem considered here is a nonhomogeneous Helmholtz equation,

$$\nabla^2 u(x, y) = h(u, x, y),$$

for which where the boundary integral equation can be written as:

$$c(\xi)u(\xi) + \int_{\Gamma} u(\eta)q^*(\xi, \eta)d\Gamma + \int_{\Omega} h[\eta, u(\eta)]u^*(\xi, \eta)d\Omega = \int_{\Gamma} q(\eta)u^*(\xi, \eta)d\Gamma.$$

The exact solution of the test problem chosen here is:

$$u = 3x^3y + 2x^2y^2 - xy^3,$$

so that the right hand side of Eq. 6 is given by:

$$h(u, x, y) = 4x^2 + 4y^2 + 12xy + 3x^3y + 2x^2y^2 - xy^3 - u.$$  

For both the particular solution method and the dual reciprocity method, fully populated linear systems of size $(M+N) \times (M+N)$ are generated. The condition number of the matrices precludes effective iterative solution, and direct solution
was used to solve the systems, resulting in an operation count which scales as \((N + M)^3\).

For the case of regular direct integration, coefficient matrices can be formed, and the linear systems can be solved directly. The diagonal dominance of the matrix results in a small condition number, and the system is amenable to iterative solution, which is generally more efficient for large systems. Here, a GMRES scheme was used. The solution times are compared in Fig. 3. As expected, the dual reciprocity and particular solution methods are solved in equal times. The solution time for the regular direct integration method is larger in all cases, however extrapolation of the curve indicates a crossover at approximately 10000 degrees of freedom. For the multipole accelerated direct domain integration, the break-even point with the dual reciprocity/particular solution methods occurs at approximately 2000 degrees of freedom. The increasing slope of the curve is a result of the increasing number of GMRES iterations required to meet the convergence criterion.

The \(L_2\)-norm error, defined as

\[
L_2 = \left[ \int_{\Omega} (u - u_{\text{exact}})^2 \, d\Omega \right]^{0.5},
\]

for systems with degrees of freedom spanning more than a decade, is also shown in Fig. 3. The dual reciprocity and particular solution methods display similar levels of accuracy, with the former being slightly more accurate, while the direct integration methods (regular and multipole accelerated) are almost two orders of magnitude more accurate.

![Figure 3: Solution time as a function of problem size, for the dual reciprocity method (◆), the particular solution method (△), the regular domain integration method (□) and the domain integration method with multipole acceleration (○) (left); Error as a function of problem size, for the dual reciprocity method (◆), the particular solution method (△), the regular domain integration method (□) and the domain integration method with multipole acceleration (○) (right).](image-url)
The assembly time for the different methods is shown in Fig. 4. Assembly time is highest for the dual reciprocity method. The assembly time for the particular solution method is approximately a factor of 4 smaller compared to the dual reciprocity method, for all problem sizes. The regular direct integration assembly time is smaller and increases more slowly than for the previous methods. As could be expected, the assembly time with multipole acceleration is the smallest, more than two orders of magnitude lower than with the dual reciprocity method.

The codes were run on a Compaq Alpha XP1000.

The assembly time with the multipole accelerated direct integration method reported in Fig. 4 includes a number of different operations, namely the calculation of moment tensors for all domain elements, the recursive subdivision of the domain, the time taken to compile far-field cluster lists and the regular assembly time for the near-field elements. Because the various steps involved in the assembly process in a multipole accelerated computation have the potential to become time-consuming, their relative importance is shown in Fig. 4.

The generation time for element moment tensors remains small compared to the other operations, until large problems are treated. This time scales as $O(P)$, where $P$ is the number of finite elements. The calculation of the near-field coefficients is always the most time-consuming operation, although the recursive domain subdivision and list compilation processes become increasingly important. The list compilation and recursive domain splitting procedures appear to become more efficient with increasing problem size.
4 Convergence

A potential problem that may be encountered when using iterative solvers is the condition number of the matrix, which is a function of its size, of the integral formulation and of the boundary conditions. To illustrate this, the calculation for various problem sizes was performed using boundary conditions which result in matrices with different condition numbers. Inspection of Eq. 7 shows that Dirichlet boundary conditions result in a first-kind Fredholm equation, while Neumann conditions produce second-kind Fredholm equations. The latter result in better conditioning of the sub-matrix of Eq. 2. In the case of mixed conditions, a larger proportion of Neumann conditions will result in better condition numbers. Solution times for the multipole accelerated direct integral method are compared in Fig. 5 for the cases where Dirichlet conditions are imposed on the side \( x = 2 \) only (of length 1) versus the case where Dirichlet conditions are imposed on the side \( y = 0 \) only (of length 2) of the same computational domain used previously. The latter boundary conditions result in worse conditioning of the sub-matrix \( A \).

![Figure 5: Solution time as a function of problem size for the multipole accelerated direct integration method: Dirichlet conditions on side \( y = 0 \) (○), Dirichlet conditions on side \( x = 2 \) (□) (left); Solution time as a function of problem size for the multipole accelerated direct integration method with Dirichlet conditions on side \( y = 0 \): by iteration of the RHS (○), and by collocation at internal nodes (□) (right).](image)

The factor of two difference in the iteration time, illustrates the effect of the condition number of sub-matrix \( A \) on the iterative solution of the entire matrix.

For some problems, it is more efficient to avoid collocation at interior nodes, and to obtain a solution by successive substitution. This method is simply a multi-dimensional fixed-point iteration, \( u = F(u) \). A comparison between the iteration times required to solve the Helmholtz problem with the two approaches is shown in Fig. 5. It is evident that the method of successive substitution is faster, by a factor of 10 at all problem sizes.

Unfortunately, the successive iterative method only converges under certain
conditions. In fact, for this particular problem it is found that convergence is only achieved when a sufficiently large portion of the boundary condition is of Dirichlet type. For example, if Dirichlet conditions are applied on the side $z = 2$, with Neumann conditions on the remaining sides, the iteration diverges, while convergence is achieved with Dirichlet conditions applied either of the sides $y = 0$, $y = 1$, or to both sides $x = 0$ and $x = 2$. Starting the iteration from an approximate solution for $u$, rather than from an initial value of 0, did not produce convergence.

A number of methods was tried to obtain convergence, including relaxation and Uzawa's method [5], which consists of applying Dirichlet conditions, and updating them based on the resultant solution until a solution that matches the desired Neumann conditions is achieved. Although a degree of convergence resulted from the application of the latter method, accurate solutions could not be obtained. More sophisticated methods, such as Newton-Raphson iterations, are feasible, but they involve the generation and inversion of a Jacobian matrix, a computationally expensive task. Newton-Raphson techniques should be effective if the solution is within the convergence radius (a situation encountered in transient simulation), while genetic algorithms may be effective in an approximate solution is not available.

5 Concluding remarks

The dual reciprocity and particular solution methods were essentially the same in terms of both CPU requirements and accuracy. The classical domain integration and domain integration by multipole expansion methods were equivalent in terms of accuracy for both example problems, reflecting the accuracy of the multipole approximation. The two domain integration methods were uniformly more accurate while typically requiring less CPU time compared to the dual reciprocity method and the particular solution method. In fact, for the two problems considered, the $L_2$-norm errors were anywhere from one to two orders of magnitude smaller for the domain integral methods.

Two methods were used to obtain a solution at the interior, namely direct collocation and successive approximation. The former provided accurate solutions for all boundary conditions. The latter were shown to converge approximately 10 times faster, but would diverge for particular sets of boundary conditions.

The main conclusions of this study are that the domain integration methods, in combination with multipole methods, are superior in terms of CPU cost, memory requirements, and accuracy compared to the two meshless methods, namely, the dual reciprocity method and particular solution method. Further, this research has demonstrated the efficacy of evaluating domain integrals in boundary element methods using multipole acceleration. With the advanced preprocessors that are available today, meshing is no longer an insurmountable problem, even with very complex geometries.
Acknowledgement

This work was supported in part by Sandia National Laboratories, a multiprogram laboratory operated by Sandia Corporation, a Lockheed-Martin Company, for the U.S. Department of Energy under Contract DE-AC04-94AL85000. The GMRES solver is used under license from CERFACS, 42 av. Gaspard Coriolis, 31057 Toulouse Cedex, France.

References


