A Galerkin implementation of the multipole expansion approach for accurate and fast solution of first and second kind Fredholm equations
Computational Fields Laboratory, Department of Electrical and Computer Engineering, Worcester Polytechnic Institute, Worcester, MA 01609, USA

Abstract
In 1983 Rohklin [9] introduced the notion of multipole expansion for the rapid solution of first and second kind Fredholm equations arising from potential theory. Recently, these ideas have been extended to 3D by the work of White, Korsmeyer [7], and others interested in rapid parameter extraction for characterization of electrical circuits.

At the same time, the Galerkin formulation of the boundary element method has recently received renewed attention. The Galerkin BEM approach can substantially increase accuracy near corner singularities, while at the same time introduce desirable convergence properties as demonstrated by Hsiao [6], and Wendland and Arnold [2].

To date the multipole expansion alternative has only been implemented and tested for collocation boundary elements. This work will discuss a Galerkin formulation of multipole expansion for two dimensional problems in electrostatics, and contrast the accuracy and computational costs of classical collocation, Galerkin BEM, collocation multipole expansion, and a Galerkin-multipole formulation.

Introduction
The solution of open field boundary value problems in electrostatics is of great interest to engineers in the field of power and high voltage engineering. Information regarding the electric field structure on and near conducting and dielectric boundaries is very useful in the design and evaluation of new devices. There are currently several methods of solving these problems, including: the finite element method, the charge simulation method, and the boundary element method. In our previous work, we have used the boundary element approach in which we model all conductor surfaces and dielectric interfaces by unknown charge distributions and enforce Dirich-
let and Neumann conditions on those boundaries while requiring that the potential be a harmonic function in the remainder of the problem domain.

In the case of two dimensional problems, the fundamental building block of the model is the line charge, represented by a point in the $x$-$y$ plane, extending to infinity in the plus and minus $z$ directions. The equation which relates the potential at a point $x$ to a line charge at a point $x_i$ is:

$$V(x) = -\frac{\rho_l}{2\pi\epsilon} \ln ||x - x_i||$$

(1)

where $\rho_l$ is the line charge density. Two dimensional boundaries can be modeled as continuous distributions of line charges. The boundaries of the two dimensional geometry are discretized into a user specified number of linear segments which we will refer to as strip elements. The equation which relates the potential at a point $x$ to the charge distribution over a strip element is:

$$V(x) = \int_0^L -\frac{\rho_s(t)}{2\pi\epsilon} \ln |x - x_i(t)|dt$$

(2)

where $t$ is the parameter that describes position along the strip, and $\rho_s(t)$ is the surface charge density. Here the charge distribution will be modeled by a linear interpolation function, with unknown charge densities defined at each end of each strip. These strip endpoints may also be used as collocation points where we enforce the Dirichlet and Neumann boundary conditions in order to solve for the unknown coefficients.

In a further enhancement of the boundary element method, we implement the Galerkin method in which the error function describing the difference between the boundary conditions and those represented by the model is forced to integrate to zero over the boundary, as opposed to satisfying them exactly at the discrete collocation points.

There are several advantages to using this method. In standard point collocation approaches, the enforcement of Neumann boundary conditions is difficult due to the lack of a clear definition of the normal to the boundary at corners and endpoints. A common solution to this problem is to model the charge distribution over these elements as a constant, taking the normal at the center of the element.

With the Galerkin method, the residual is integrated over the length of the element, and the normal is unambiguous. This allows for the use of more physical higher order representations of the charge distribution along dielectric interfaces. Furthermore, Galerkin is known to provide superior results for singular charge distributions on Dirichlet boundaries [3].

An important disadvantage of the Galerkin method is the increased computational cost. In the case of numerical direct integration, the minimum
increase in matrix assembly time is a factor of three as compared to point collocation. The practical increase in accuracy, particularly for Dirichlet problems, can be marginal, and often the cost of the better answer is prohibitive. It is the purpose of this work to demonstrate an alternative to constructing the influence matrix that substantially reduces the cost of assembly. Consequently, the Galerkin formulation with its attractive attributes does become a sensible option, particularly for boundaries with singular source distributions and for boundaries defining multi-dielectric regions.

**Derivation**

In both point collocation and Galerkin formulations, the problem size is specified by the number of boundary elements and nodes. Using direct integration to construct the influence matrix requires evaluation of boundary conditions at \( N \) nodes, due to the charge distribution which is specified at the same \( N \) nodes. Clearly, the number of operations involved in forming the matrix is a constant multiplied by \( N^2 \), which can become computationally prohibitive for large numbers of elements. In this work we will discuss an algorithm which can substantially reduce the constant, thereby reducing the time required to form the matrix.

Multipoles take advantage of the fact that the importance of the details of the geometric complexities of a charge distribution are smeared as a function of distance. In fact, at a great distance, a charge distribution may be acceptably modeled as being concentrated at a single point charge (or line charge) located within the distribution. This type of model is known as the monopole approximation of the charge distribution, and the monopole is the first term in the infinite multipole expansion series which completely describes the effect of the distribution when viewed at a sufficient distance. We shall seek to model all sufficiently separated charge distributions as truncated multipole expansions for the purpose of building the Green matrix. In general, we will use a second order multipole approximation which, in addition to the monopole term, includes a dipole term and a quadrupole term.

We begin by restating the expression for potential due to a line charge in complex form:

\[
V(\mathbf{x}) = -\frac{\rho_i}{2\pi\varepsilon} \Re(\ln(\mathbf{x} - \mathbf{x}_i))
\]  

(3)

This expression relates potential due to a line charge at a position \( \mathbf{x}_i \) relative to the origin, however we require an expression in terms of position relative to an arbitrary point \( \mathbf{x}_0 \). We therefore employ the coordinate translation:

\[
\mathbf{x}' = \mathbf{x} - \mathbf{x}_0
\]  

(4)

to obtain:

\[
V(\mathbf{x}) = -\frac{\rho_i}{2\pi\varepsilon} \Re(\ln(\mathbf{x}' - (\mathbf{x}_i - \mathbf{x}_0)))
\]  

(5)
Note that the natural log can be series expanded:

$$\ln(\hat{x} - \hat{x}_i) = \ln \hat{x} - \sum_{m=1}^{\infty} \frac{1}{m} \left( \frac{\hat{x}_i}{\hat{x}} \right)^m$$  \hspace{1cm} (6)

If we replace \( \hat{x} \) with \( \hat{x}' \) and \( \hat{x}_i \) with \( (\hat{x}_i - \hat{x}_0) \) in the last equation, we can apply the series expansion for the logarithm to equation (5) to obtain:

$$V(\hat{x}) = -\frac{\rho_i}{2\pi \epsilon} \Re \left[ \ln \hat{x}' - \sum_{m=1}^{\infty} \frac{1}{m} \frac{\rho_i(\hat{x}_i - \hat{x}_0)^m}{(\hat{x} - \hat{x}_0)^m} \right]$$  \hspace{1cm} (7)

Distributing the charge density, using the translation substitution, and factoring the series expression, we obtain:

$$V(\hat{x}) = -\frac{1}{2\pi \epsilon} \Re \left[ \rho_i \ln(\hat{x} - \hat{x}_0) - \sum_{m=1}^{\infty} \frac{1}{m} \frac{\rho_i(\hat{x}_i - \hat{x}_0)^m}{(\hat{x} - \hat{x}_0)^m} \right]$$  \hspace{1cm} (8)

As seen in Figure 2, this is a multipole expansion, centered about a point \( \hat{x}_0 \), for the potential at a point \( \hat{x} \) due to a line charge at a point \( x_i \).

Converting line charges over to a strip distribution with linearly varying charge density and parameterized length extending from \(-1\) to \(1\), we obtain:

$$V(\hat{x}) = -\frac{1}{2\pi \epsilon} \Re \left[ a_0 \ln(\hat{x} - \hat{x}_0) - \sum_{m=1}^{\infty} \frac{\hat{a}_m}{(\hat{x} - \hat{x}_0)^m} \right]$$  \hspace{1cm} (9)

where \( a_0 = L_i \frac{\rho_i- + \rho_i+}{2} \)

$$\hat{a}_m = \frac{1}{m} L_i \left[ \rho_i- \int_{-1}^{1} (1 - t) \left( \frac{\hat{x}_i- - \hat{x}_i-}{2} t + \frac{\hat{x}_i+ + \hat{x}_i+}{2} - \hat{x}_0 \right)^m dt \right. + \rho_i+ \int_{-1}^{1} (1 + t) \left( \frac{\hat{x}_i+ - \hat{x}_i-}{2} t + \frac{\hat{x}_i+ + \hat{x}_i+}{2} - \hat{x}_0 \right)^m dt \right]$$

where \( L_i \) is the length of strip \( i \), and \( \rho_i- \) and \( \rho_i+ \) are the charge densities at the endpoints of the strip element \( i \).

If we take a superposition sum of \( N \) strip distributions we obtain:

$$V(\hat{x}) = -\frac{1}{2\pi \epsilon} \Re \left[ a_0 \ln(\hat{x} - \hat{x}_0) - \sum_{m=1}^{\infty} \frac{\hat{a}_m}{(\hat{x} - \hat{x}_0)^m} \right]$$  \hspace{1cm} (10)

where \( a_0 = \sum_{i=1}^{N} L_i \frac{\rho_i- + \rho_i+}{2} \)
\[ \hat{a}_m = \frac{1}{m} \sum_{i=1}^{N} L_i \int_{-1}^{1} \left( \rho_i - \int_{-1}^{1} \left( \frac{\hat{x}_{i+} - \hat{x}_{i-} - \hat{x}_{i-} + \hat{x}_{i+}}{2} - \hat{x}_0 \right)^m dt + \rho_i + \int_{-1}^{1} (1 + t) \left( \frac{\hat{x}_{i+} - \hat{x}_{i-} - \hat{x}_{i-} + \hat{x}_{i+} - \hat{x}_0}{2} \right)^m dt \right) \]

Now consider the format of these expressions. First, the infinite series may be truncated to a finite order \( K \), the value of which is dependent upon the implementation. Using this expression to evaluate the potential at many distinct points due to the charge distribution represented by the multipole, only the evaluation coordinate \( \hat{x} \) changes, and the coefficients \( a_0 \) and \( a_m \) need be evaluated only once for each of the \( N \) strip elements. Interestingly, many interactions will not meet the distance criteria necessary and will require direct evaluation. In order to further decrease the cost of matrix assembly we can gain further efficiency by carefully choosing the level from which we take the expansions.

In order to achieve this end, we must be able to shift the center of a multipole region from one point to another. Employing a simple change of coordinate variable, we easily obtain an expansion of exactly the same form, referenced to a new center point. In this manner, we are able to obtain a multipole expansion for any square region from the expansions of the four smaller squares of which it is composed. The geometric criteria that determines when we may use a multipole to represent a distribution of charge is simply that of the well separated region. The expansion for the potential at a point \( \hat{x} \) due to a multipole representation of a region of charge bounded by a circle of radius \( R \) centered about \( x_0 \) can be shown to converge rapidly with increasing order of the expansion for all points some minimum distance from \( x_0 \). This allows us to determine the largest region of charge that may be represented as a multipole for any given evaluation point. Notice that when we combine any four regions into a single region in this manner, the radius of the new region is twice that of any of the individual regions from which it is built.

For Neumann boundary conditions, we must have the ability to evaluate the field due to a multipole. If we take the gradient of equation (9), we obtain:

\[
E(\hat{x}) = \frac{1}{2\pi \varepsilon} \mathcal{R} \left( \frac{a_0}{\hat{x} - \hat{x}_0} + \sum_{m=1}^{\infty} \frac{m \hat{a}_m}{(\hat{x} - \hat{x}_0)^{m+1}} \right) a_x \\
+ \frac{1}{2\pi \varepsilon} \mathcal{R} \left( \frac{j a_0}{\hat{x} - \hat{x}_0} + \sum_{m=1}^{\infty} \frac{j m \hat{a}_m}{(\hat{x} - \hat{x}_0)^{m+1}} \right) a_y
\]  

(11)

With this expression, we are now able to find the field, and therefore
the normal field, at any point. Details of the derivation for both first and second kind Fredholm operators using collocation and Galerkin boundary elements can be found in [1].

Implementation

A mesh, and multipole coefficients for all regions at all levels of the mesh, must be created before matrix assembly can begin. In contrast to the direct integration approach in which few preliminary operations are required, this additional overhead which may seem problematic in terms of accelerating matrix assembly, however, it will be shown to be negligible.

We begin by defining the “root region” which is the smallest square that completely encloses the problem domain. The root region is then subdivided into four “child” squares. At this point, each child is examined to determine how many elements it contains. If any of these regions contain more than a predefined default number of elements, it too is subdivided, and the process repeats recursively until the entire problem domain is subdivided into smaller “lowest levels,” as in Figure 3.

Having fully partitioned the domain, we are now ready to create the multipole coefficients. Begin by selecting the root region. If this active region has children, each child is activated in turn, and the process repeats recursively until a childless (or lowest level) region is selected. When a lowest level region is activated, multipole coefficients for that cell are generated. At this point, the algorithm returns to the parent of this first lowest level region, and then down to the other children. When the algorithm returns from the fourth child, it translates all four multipoles from the center of the children to the center of the parent. When this recursive function terminates, all regions at all levels possess a multipole expansion.

Each lowest level multipole contains five coefficients for each strip endpoint in that cell. As these are integrated into the parent regions, the number of coefficients is maintained. Consequently, parent multipoles contain maps to each strip element contained within their respective regions. For a problem with 1000 unknowns, the capacitance matrix is 8 megabytes, while the vector representing the multipole map to strip elements is on the order of 1 megabyte, depending upon the number of levels.

A sequence of translations to the center of the highest level parent enables rapid calculation of field point parameters. Clearly, the multipole coefficients are determined one time; additionally, each field point distance is calculated once per multipole, and therein lies the advantage of this approach over direct integration. The capacitance matrix is solved using Gaussian elimination.
Having completed preliminary data generation, matrix assembly may begin. As before, the root region is searched recursively until all lowest level regions are individually selected. For the collocation implementation, each node contained within the region is used as a collocation field point. For the Galerkin implementation, each strip element in the region is individually selected to be the element over which the Galerkin integration is performed.

Once again, the root region is activated and examined, this time to determine if it may be used as the source region. If the region center is sufficiently distanced from the evaluation element, the multipole expansion coefficients for that region may be used to generate the matrix coefficients at the field point. If the active region is not sufficiently far away, the children of the active region are recursively searched until a region is encountered such that the distance is sufficient (as compared to that region's radius). If no such region is encountered, the recursive search will ultimately reach a lowest level region whose matrix coefficients are then generated using direct integration. This process forces matrix assembly using the largest possible multipoles.

When a region is activated that may be used as a multipole region, matrix coefficients are determined utilizing the appropriate relationships as previously described. That is, the expression for potential or field due to a multipole is used for Dirichlet and Neumann field elements, respectively. If the Galerkin method is being employed, the integration is performed using three-point Gaussian quadrature. Remember, all Neumann boundaries are processed by the Galerkin method.

Comments on the Cost of the Algorithm

The process of forming a matrix fundamentally requires order $N^2$ operations, where $N$ is the size of the matrix. This multipole algorithm does not escape that restriction, however it redistributes much of the computational work such that only a fraction of the calculations need to be performed in order $N^2$, and most of the work is performed in only order $N$. For the traditional direct integration approach there is a constant number of calculations which need to be performed to fill each entry in the matrix; the cost of filling $N^2$ matrix entries is $k \cdot N^2$, where $k$ is determined by the specific computer hardware used.

The cost of filling a matrix using multipole expansions is more difficult to quantify, since the actual number of operations performed depends not only on how many strips are in the problem, but also on the layout of the geometry, the mesh size, and the minimum distance criteria. The equation describing the cost of filling a single row of the matrix using a combination
of direct integration and multipole approximation is:

\[
\text{Row cost} \approx k_d \cdot n + k_m \cdot (N - n)
\]  

(12)

where \(k_d\) and \(k_m\) are the cost of computing one entry of the matrix using direct integration and the multipole algorithm, respectively, and \(n\) is the number of entries in the matrix row which are evaluated with direct integration. Assume that \(n\) is the same for all rows of the matrix. Using our algorithm \(k_m\) is substantially smaller than \(k_d\), by a factor that ranges from approximately 12 to 30.

Equation (12) can be rewritten as:

\[
\text{Row cost} \approx k_m \cdot N + (k_d - k_m) \cdot n
\]  

(13)

and the cost of filling the entire matrix of \(N\) rows becomes:

\[
\text{Cost} \approx k_m \cdot N^2 + (k_d - k_m) \cdot n \cdot N
\]  

(14)

In the case where all of the matrix entries are computed using direct integration \((n = N)\), the cost of building the matrix becomes:

\[
k_d \cdot N^2
\]

In contrast, the cost for filling the matrix using only the multipole approximation \((n = 0)\) would be:

\[
k_m \cdot N^2
\]

Notice that in this extreme case the cost of assembling the matrix is as much as \(k_d/k_m\) times less than using only direct integration. In practice, \(n\) is somewhere between 0 and \(N\), closer to 0 than \(N\) when \(N\) is large, and depends on the geometry and discretization of the problem.

The cost of the multipole algorithm must fundamentally follow equation (14). In the general case, as \(N\) increases, \(n\) also increases, but the ratio \(n/N\) goes down. When \(N\) is very small most of the coefficients in the influence matrix must be computed using direct integration. As \(N\) increases, the term \((k_d - k_m) \cdot n \cdot N\) does not increase as quickly as \(k_m \cdot N^2\), and the cost of forming the matrix becomes dominated by \(k_m \cdot N^2\).

Results

To illustrate the quality of the multipole approximation, as well as to demonstrate its speed advantage over direct integration, two geometries were simulated. The first, depicted in figure 1, consists of two perfectly conducting cylindrical conductors of infinite length. The left cylinder has a potential of \(-1\) Volt while the one on the right is at 1 Volt. The problem was simulated using both the direct integration and the multipole approximation
algorithms for the point collocation and Galerkin element methods. The number of strips used to represent the geometry in this analysis ranges from 40 to 1000, and it was chosen because it possesses an analytical solution describing charge along the boundary.

Figure 4 shows the integrated error over all the boundaries of the charge density computed by the four different solution options, as calculated by:

$$\rho_{error} = \frac{\sqrt{\int_{\Gamma} (\rho - \rho_{ex})^2 \, d\Gamma}}{\int_{\Gamma} \rho_{ex} \, d\Gamma}$$

(15)

where $\Gamma$ represents the entire boundary, $L$ is the length of the entire boundary, and $\rho$ and $\rho_{ex}$ are the computed solution and the exact analytic solution, respectively. Surprisingly, the error rises slightly for the multipole solutions as $N$ increases, however, the slope is very small and the solution maintains excellent agreement with the analytic solution. The increasing error may be attributable to the multipole approximation.

The error in potential integrated over both boundaries for the same geometry as given by:

$$V_{error} = \frac{\sqrt{\int_{\Gamma} (V - V_{ex})^2 \, d\Gamma}}{\int_{\Gamma} V_{ex} \, d\Gamma}$$

(16)

is shown in figure 5, where $V$ and $V_{ex}$ are the computed solution and the given boundary conditions, respectively. The error for the multipole algorithm is larger than for direct integration, but the difference is less than one hundredth of one percent. Notice that in this simple problem, increasing the number of strip elements beyond 300 does not further reduce the error significantly.

The second geometry, shown in figure 6, consists of two perfectly conducting parallel plates with a dielectric slab adjoining one of the conductors. The upper plate has a potential of 1 Volt and the lower plate $-1$ Volt. The dielectric constant of the dielectric slab is 8. There is no exact analytic solution for charge distribution for this problem, and therefore we could not compare the computed charge distribution directly. The error in potential for this problem is given in figure 7. The magnitude of the error is larger than in the previous example because this problem contains sharp corners, and is not significantly reduced by using more than 300 strip elements. Notice that the Galerkin multipole error is virtually indistinguishable from the Galerkin direct integration error.

Figure 8 shows the total amount of time required to assemble the influence matrix using the different algorithms, as well as the time to solve
the matrix using Gaussian elimination. As $N$ increases, the cost of using Galerkin elements with direct integration becomes prohibitively expensive. In contrast, the added cost of Galerkin elements using the multipole algorithm remains negligible with respect to the total solution time of the problem without losing accuracy. The multipole algorithm makes it possible to use Galerkin elements in a cost-effective way for large problems.

Figure 9 shows the times to perform the various phases of the multipole algorithm. The combined time of performing the preliminary steps of building the mesh, creating node lists, and calculating the multipole coefficients is insignificant compared to the time assembling the matrix. The total time to build the matrix is dominated by the matrix filling process, as described in the section “Comments on the Cost of the Algorithm.”

Conclusions

Galerkin boundary elements are clearly superior to classical collocation formulations for Neumann boundary conditions, and are comparable to collocation for smooth Dirichlet problems. However, the cost of Galerkin using direct integration can be prohibitive, and it is not always clear whether one should simply employ a larger number of collocation elements to achieve similar accuracy.

The point of this paper, given the attractive features of the Galerkin algorithm but its prohibitive cost, is that we can sensibly employ the Galerkin formulation if the coefficient matrix is built via the multipole alternative.

For practical problems on the order of 400 unknowns, the time to assemble the (collocation) matrix and the time to solve the full matrix using Gaussian elimination are similar. Now, using multipole expansion, the time to assemble the Galerkin matrix has been reduced by a factor of five with no serious degradation of quality, thus making this formulation competitive once again. Indeed, it is still more expensive than its collocation counterpart, however, overall assembly time can now be considered relatively inexpensive.

While there are apparently ways in which the order $N^2$ operations could be expedited, White [8] has demonstrated that coupling the assembly process to an iterative solver like GMRES can effectively reduce the order of operations to order $N$. This work is distinguished from that and similar efforts only insofar as we have demonstrated that multipoles provide a sensible alternative for implementing the Galerkin formulation.
References


Figure 1: An example boundary value problem: two perfectly conducting cylindrical conductors of infinite length with a potential difference between them. The objective is to calculate the charge distribution along the conductor-dielectric interfaces. In order to solve the problem numerically, the boundaries are discretized into strip charge elements.

Figure 2: Region containing a set of elements which may be represented by a multipole expansion. For any point outside the circle of radius $R$, the multipole expansion located at $\mathbf{x}_0$ will produce the same electric field as the superposition sum of the individual contributions from all elements. When a truncated multipole expansion is used, the error is limited if the field point is outside the dashed circle.
Figure 3: A meshed plot of the geometry shown in figure 1. The mesh is refined until there are no more than a predetermined number of strips in any given region. In areas where there are many strips, the sizes of the regions are smaller.

Figure 4: Integrated error over all boundaries in the approximate charge density as compared to the analytic solution for the geometry of figure 1. The direct integration approach and the multipole algorithm are applied to both the point collocation and Galerkin methods, resulting in four curves. The ordinate scale is in hundreths of percent, nonetheless the slow increase in multipole error is surprising.
Figure 5: Integrated error over both boundaries of the potential calculated from the computed charge density for the geometry of figure 1. Although the error shown for the multipole expansion is higher than the direct integration error, the results show excellent agreement to the imposed boundary conditions.

Figure 6: Another boundary value problem: two perfectly conducting parallel plates with a potential between them, with a dielectric slab adjoining one of the conductors. This problem includes both Dirichlet and Neumann boundary conditions.
Boundary Elements

Error in Potential for Parallel Plate Geometry

- Collocation Direct
- Galerkin Direct
- Collocation Multipole
- Galerkin Multipole

Figure 7: Integrated error over all conducting boundaries of the potential calculated from the computed charge density for the geometry of figure 6. The Galerkin multipole solution converges more quickly than the collocation multipole solution. The difference between the Galerkin direct integration error and the Galerkin multipole error is indistinguishable.

Total Time to Build Matrix

Figure 8: The amount of time needed to build the influence matrix as a function of the number of strips in the problem. The multipole alternative is always faster than direct integration, even for our relatively inefficient implementation. The additional overhead imposed by the Galerkin formulation using multipoles is negligible compared to the time required by a Gaussian solver for 600 unknowns or more.
Figure 9: The amount of time spent in each phase of filling the matrix using the Galerkin, multipole algorithm. The preliminary steps of forming the mesh, creating nodes lists, and calculating the multipole coefficients are insignificant compared to the time spent actually calculating the matrix coefficients.