Local Defect Correction for the Boundary Element Method

G. Kakuba, R.M.M. Mattheij and M.J.H. Anthonissen

Eindhoven University of Technology
Department of Mathematics and Computer Science
P. O. Box 513, 5600MB Eindhoven, The Netherlands
g.a.kakuba@tue.nl

Abstract

We present an efficient way to implement the Boundary Element Method (BEM) to capture high activity regions in a boundary value problem. In boundary regions where accuracy is critical, like in adaptive surface meshes, the method of choice is Local Defect Correction (LDC). We formulate the method and demonstrate its applicability and reliability by means of an example. Numerical results show that LDC and BEM together provide accurate solutions with less computational requirements given that BEM systems usually consist of dense matrices.

1 Introduction

In the boundary element method, the solution of a function on a given domain is expressed as an integral equation in terms of its values and normal derivatives at the domain boundary. The boundary conditions are either Dirichlet or Neumann or both in
the case of a mixed boundary conditions problem. Sometimes the boundary contains small regions of high activity. For example, for the Impressed Current Cathodic Protection (ICCP) Systems solved in [Kakuba, 2005, Lenaers, 2006], the boundary consists of small active regions there referred to as electrodes. In [Pozrikidis, 1992], a solution to a three-dimensional flow problem with a local region of high activity is also presented. In all these cases uniform grids have been used. For better accuracy with such localised regions of high activity, one usually has to employ a rather fine grid, either globally or as a composite grid. As discussed in [Pozrikidis, 2002], one way to circumvent local decreases in the accuracy of the BEM that occurs due to this local high activity is to use a high density of boundary elements globally. However, this results into extremely large systems that are computationally demanding especially in the case of BEM where matrices are usually dense. The advantages that BEM may have over FEM and FDM are then annihilated. In this paper we show how to circumvent this by incorporating Local Defect Correction (LDC) into BEM in the case of localised regions that require a high density of elements.

In [Anthonissen, 2001] LDC techniques are well analysed for finite difference and finite volume methods and some applications are also discussed. In [Minero et al., 2006] LDC techniques for time-dependent problems are discussed. In this article we extend the LDC method for use with BEM. In the LDC method a fine grid solution is used to overall improve the coarse grid approximation. This is achieved through a so-called defect correction in which the fine grid solution is used to approximate the local discretisation error of the coarse grid. We give a brief development of the BEM and explain local defect correction. We combine BEM and LDC in an algorithm that we shall call a BEM-LDC algorithm. Numerical experiments are carried out using a suitable example and results
discussed. The results show that we can reduce the complexity of the solution method by at least a factor two while keeping a good accuracy. In the particular case of the problems that have been solved in [Kakuba, 2005, Lenaers, 2006], the use of BEM-LDC would reduce the complexity by at least a factor 40!

The paper is built up as follows: in the next section, Section 2, we give a brief description of BEM and explain the LDC algorithm for BEM. In particular, in Section 2.1, a brief account of BEM formulation is given. The local problem and local defect are explained in Sections 2.2 and 2.3 respectively. In Section 2.4, we describe defect correction and present the BEM-LDC algorithm. Section 3 brings all the previous sections together by discussing some numerical experiments and their results. The last section, Section 4, will be devoted to conclusions.

2 Boundary Element Method with Local Defect Correction

In this section we explain the development of the BEM-LDC algorithm. A brief description of the BEM formulation is presented. We give an account of the LDC technique and then show how we combine BEM and LDC together to get a better solution to a boundary value problem.

2.1 Boundary Element Method, a First Approximation of the Solution

Let us consider the exterior potential problem governed by the Laplace equation and given by the boundary value problem
\[
\begin{equation}
\begin{cases}
\nabla^2 u(r) = 0 & r \in \Omega, \\
u(r) = g_1(r) & r \in \partial \Omega_1, \\
\frac{\partial u}{\partial n}(r) := q(r) = g_2(r) & r \in \partial \Omega \setminus \partial \Omega_1,
\end{cases}
\end{equation}
\]

where \(\partial \Omega\) denotes the boundary of the problem domain \(\Omega\). For a smooth boundary, the integral equation is, see [Kakuba, 2005], [Paris and Canas, 1997], [Pozrikidis, 1992],

\[
\frac{1}{2} u(r) = \int_{\partial \Omega} u(r') \frac{\partial v}{\partial n}(r'; r) \, dS(r') - \int_{\partial \Omega} v(r'; r) \frac{\partial u}{\partial n}(r') \, dS(r'), \quad r \in \partial \Omega.
\]

The surface \(\partial \Omega\) is divided into \(N\) elements; in this paper we use constant triangular elements whose accuracy has been shown in [Lenaers, 2006] to be as good as that of triangular linear elements except for very special cases. Thus, on each of the elements \(i = 1, 2, \ldots, N\), we assume that the functions \(u\) and \(q\) remain constant. We introduce the notation \(u^i, q^i\) and \(\partial \Omega_j\) to denote \(u(r_i), q(r_i)\) and the \(j\)-th triangle respectively. Discretising eq (2) and letting the fixed point \(r\) to be the collocation point at element \(i\), we get

\[
\frac{1}{2} u^i - \sum_{j=1}^{N} u^j \int_{\partial \Omega_j} \frac{\partial v}{\partial n}(r'; r_i) \, dS(r') = - \sum_{j=1}^{N} q^j \int_{\partial \Omega_j} v(r'; r_i) \, dS(r').
\]

Writing (3) for all the elements \(i = 1, 2, \ldots, N\), the resulting equations can be written in matrix form as

\[
Hu = Gq,
\]

where

\[
H_{ij} = \tilde{H}_{ij} - \frac{1}{2} \delta_{ij}, \quad \tilde{H}_{ij} = \int_{\partial \Omega_j} \frac{\partial v}{\partial n}(r'; r_i) \, dS(r').
\]
and

\[ G_{ij} = \int_{\partial\Omega_i} v(r'; r_i) \, dS(r'). \]  \hspace{1cm} (6)

Applying boundary conditions to (4) results into the system

\[ A^H x^H = b^H \]  \hspace{1cm} (7)

where \( x^H \) contains the unknown values of \( u \) and \( q \). Solving this system gives us an initial approximation \( x^H_0 \). The superscript \( H \) denotes the solution on a coarse grid (grid size \( H \)).

Now assume that the continuous solution \( u \) of (1) has a high activity region in some (small) part of the domain boundary. This high activity of \( u \) may be captured by discretising (2) on a composite grid. To this end, we choose \( \partial\Omega_1 \subset \partial\Omega \) such that the high activity of \( u \) is contained in \( \partial\Omega_1 \). In \( \partial\Omega_1 \), we choose a local fine grid (grid size \( h \)), which we denote by \( \partial\Omega_1^h \). The fine grid is chosen such that \( \partial\Omega^H \cap \partial\Omega_1 \subset \partial\Omega_1^h \), i.e., element nodes of the global coarse grid that lie in the area of refinement are nodes of local fine grid too.

When (2) has been discretised and solved on a coarse grid, and when an area of the coarse grid has been refined and a local solution has been calculated on the finer grid, we can use the local fine grid to update the coarse grid approximation.

Suppose \( x^* \) is the exact solution, then substitution in (7) would give

\[ A^H x^* = b^H + d^H \]  \hspace{1cm} (8)

where the defect \( d^H \) is a vector of the local discretisation errors. If we would know the values of the defect \( d^H \), we could add them to the right hand side of (7) and solve this system of equations to find the exact values of \( u \) and \( q \) in (4).
2.2 The Local Problem

Since the exact solution is normally unknown, we cannot calculate \(d^H\) in eq (8). What we can do though is to use the approximation calculated on the local fine grid to estimate \(d^H\) for those grid elements of the global coarse grid that lie within \(\partial\Omega_1\). To formulate the local problem, we form a composite grid wherein we have a fine grid in the local region whereas we maintain the coarse grid elsewhere. Therefore, in eq (7), we now denote the operator \(A^H\) as \(A^{H,h}\) and the vectors \(x^H\) and \(b^H\) are split in such a way that we can write,

\[
A^{H,h} \begin{pmatrix} x^H_i \\ x^h_i \end{pmatrix} = \begin{pmatrix} f^H_i \\ f^h_i \end{pmatrix}. \tag{9}
\]

Let us split the operator \(A^{H,h}\) so that eq (9) can be written as

\[
\begin{pmatrix} A^H_c & B^{H,h}_{c,l} \\ C^{H,h}_{l,c} & A^h_l \end{pmatrix} \begin{pmatrix} x^H_c \\ x^h_l \end{pmatrix} = \begin{pmatrix} f^H \\ f^h_l \end{pmatrix}. \tag{10}
\]

From eq (10) we extract two equations:

\[
A^H_c x^H_c + B^{H,h}_{c,l} x^h_l = f^h_l, \tag{11a}
\]

\[
A^H_c x^H_c + B^{H,h}_{c,l} x^h_l = f^H. \tag{11b}
\]

Having added more elements in the local region, see Figs. 1 and 2, we rewrite eq (11a) in the form

\[
A^h_l x^h_l = f^h_l - C^{H,h}_{l,c} x^H_c. \tag{12}
\]

We split the initial coarse grid solution \(x^H_0\) in the form \(\begin{pmatrix} x^H_{c,0} \\ x^H_{l,0} \end{pmatrix}\) where \(x^H_{c,0}\) is the solution outside the local region and \(x^H_{l,0}\) is the solution in the local region. We then substitute \(x^H_{c,0}\) for \(x^H_c\) in eq (12) and hence obtain the following system of equations for the local problem:

\[
A^h_l x^h_{l,0} = f^h_l - C^{H,h}_{l,c} x^H_{c,0} = b^h_l. \tag{13}
\]
Figure 1: A composite grid discretisation: The thick rectangle encloses the active region, the large dots indicate nodes that belong to $\partial \Omega_i^H$. The thick mesh shows the triangles of the coarse grs that belong to the active region.

Figure 2: At each refinement step, a coarse triangle is divided into four.
Thus we form a local problem in such a way that outside the local region, the solution is known and is the global coarse grid solution. This results in a smaller system that is cheaply solved to yield the local problem initial solution $x_{l,0}^h$.

### 2.3 The local defect

In the constant elements discretisation, the values of the functions on a particular element are associated with the centroids of the triangles which are called nodes. Let $k$ denote a triangle node. Let us partition the elements of the coarse grid as

$$\partial \Omega^H = \partial \Omega^H_l \cup \partial \Omega^H_c,$$

where $\partial \Omega^H_l := \{ k \in \partial \Omega^H | k \in \partial \Omega_h \}$ and $\partial \Omega^H_c := \partial \Omega^H \setminus \partial \Omega^H_l$. If $u^*$ and $q^*$ denote the exact values of $u$ and $q$ respectively, then we would have the local discretisation errors given by, from (3),

$$d_i := \sum_{j=1}^{N} u_j^* \int_{\partial \Omega_j} \frac{\partial v}{\partial n}(r'; r_i) \, dS(r')$$

$$- \sum_{j=1}^{N} q_j^* \int_{\partial \Omega_j} v(r'; r_i) \, dS(r') - \frac{1}{2} u_i^*$$

$$= \sum_{j=1}^{N} \left( \hat{H}_{ij} - \frac{1}{2} \delta_{ij} \right) u_j^* - \sum_{j=1}^{N} q_j^* G_{ij}$$

$$= \sum_{j=1}^{N} H_{ij} u_j^* - \sum_{j=1}^{N} q_j^* G_{ij}, r_i \in \partial \Omega^H. \quad (15)$$

Applying boundary conditions, (15) yields

$$d_i = \sum_{j=1}^{N} A_{ij} x_j^* - b_i, \quad r_i \in \partial \Omega^H. \quad (16)$$

If we write eq(16) for all $r_i \in \partial \Omega^H_l$, we obtain the following local defect on $\partial \Omega^H_l$, in vector form

$$d_l^H = A_{id}^H x_d^H - b_l^H.$$  

(17)
Here, $x_{ci}^H$ is the projection on $\partial \Omega^H$ of the exact solution. We can partition the matrix $A_{ci}^H$ into the form

$$A_{ci}^H = \begin{pmatrix} C_{ic}^H & A_i^H \end{pmatrix},$$

so that we have

$$d_i^H = C_{ic}^H x_c^H + A_i^H x_i^H - b_i^H,$$

(19)

where $x^H = (x_i^H, x_c^H)$ and $x_i^H$ and $x_c^H$ are the projections of $x^H$ onto $\partial \Omega_i^H$ and $\partial \Omega_c^H$ respectively.

Since we do not know the exact solution $x^*$, we cannot calculate $d_i^H$ using (19). What we can do though, is to use the approximation $x_{i,0}^H$ calculated on the local fine grid to estimate $d_i^H$. Using (19), we find

$$d_i^H = C_{ic}^H x_c^H + A_i^H x_i^H - b_i^H$$

$$\approx C_{ic}^H x_{c,0}^H + A_i^H R_{ih} x_{i,0}^H - b_i^H$$

$$:= d_{i,0}^H.$$

(20)

Here we have introduced the operator $R_{ih}$ which is the restriction from $\partial \Omega_i^h$ onto $\partial \Omega_i^H$.

### 2.4 The Defect correction and the BEM-LDC Algorithm

In light of (19) and (20), we partition the matrix $A_i^H$ in (7) into the form

$$A_i^H = \begin{pmatrix} A_{ci}^H & B_1 \\ B_2 & A_i^H \end{pmatrix}.$$

(21)

The vectors $x_i^H$ and $b_i^H$ are also partitioned accordingly so that (7) can be rewritten as

$$\begin{pmatrix} A_{ci}^H & B_1 \\ B_2 & A_i^H \end{pmatrix} \begin{pmatrix} x_c^H \\ x_i^H \end{pmatrix} = \begin{pmatrix} b_c^H \\ b_i^H \end{pmatrix}.$$
Using (20) we find an estimate of the local discretisation error of the coarse grid discretisation at all points of $\partial \Omega^H_l$. Therefore, we can update the coarse grid approximation by placing the estimate in (20) at the right hand side of the coarse grid eq (7) or (22). This leads to the coarse grid correction step to find $x^H_1, i = 1$, on the coarse grid:

$$
A^H x^H_1 = \begin{pmatrix}
    b^H_c \\
    b^H_l + d^H_{l,i-1}
\end{pmatrix} = \begin{pmatrix}
    b^H_c \\
    C^H_{l,c} x^H_{c,0} + A^H_{l} R^{H,h} x^h_{l,0}
\end{pmatrix}. \quad (23)
$$

Because (23) incorporates estimates of the local discretisation error of the coarse grid discretisation, the new solution $x^H_1$ is assumed to be more accurate than $x^H_0$. Hence a solution based on the new approximation $x^H_1$ provides a better boundary condition for the local problem. A better solution on the local fine grid can be found as before by solving (13) with $x^h_{1,1}$. In summary we have the following algorithm:

**Algorithm 1, BEM-LDC Algorithm**

- **Initialisation**
  - Solve the basic coarse grid problem (7)
  - Solve the local fine grid problem (13) with $i = 0$.

- **Iterations $i = 1, 2, \ldots$**
  - Solve the updated coarse grid problem (23)
  - Solve the local fine grid problem (13)
3 Numerical experiments

Consider the following exterior potential problem:

\[
\begin{align*}
\nabla^2 u(r) &= 0 & \quad r \in \Omega = \mathbb{R}^3 \setminus \Omega^c, \\
u(r) &= \frac{1}{r} & \quad r \in \partial \Omega \setminus \partial \Omega_1, \\
\frac{\partial u}{\partial n}(r) := q(r) &= \frac{\partial}{\partial n} \left( \frac{1}{r} \right) & \quad r \in \partial \Omega_1,
\end{align*}
\]

where

\[r = ||r - r_0||, \quad r = (x, y, z),\]

\[\Omega^c = \{ r : r \in [0, 4] \times [0, 10] \times [0, 6] \},\]

\[\partial \Omega_1 = \{ r : x = 4.0 \}.\]

Let \(e = u - \hat{u}\), where \(\hat{u}\) is the numerical solution and \(u\) the exact solution, denote the error vector in estimating \(u\). In (25), let \(r_0 = (3.7, 5.0, 3.0)\). Solving the problem on the coarse grid using BEM with a uniform distribution of elements, we find that \(||e_0||_\infty = 0.5544\) for \(H = 1.0\) (496 elements) whereas \(||e_0||_\infty = 0.2589\) for \(H = 0.5\) (1984 elements), where \(e_0 = u - \hat{u}_0\). Using the BEM-LDC algorithm with \(h = 0.5\), the updated coarse grid solution error is \(||e_1||_\infty = 0.1385677\). This is a good promise.

Fig. 2 shows how we refine in the local active region, a coarse grid triangle is divided into four triangles by dividing the lengths of the two short sides by two and then connecting the resulting points through a triangular path. Fixing \(r_0 = (3.7, 5.0, 3.0)\) creates a region of high activity on the surface \(\partial \Omega\) around the point \(r = (4.0, 5.0, 3.0)\). Let the local active region be

\[\partial \Omega_1 = \{(x, y, z) : x = 4.0, 4.0 \leq y \leq 6.0, 2.0 \leq z \leq 4.0 \} .\]

To further analyse results of the BEM-LDC algorithm, we have increased the activity by setting \(r_0 = (3.9, 5.0, 3.0)\) solved the boundary value problem (24). The results obtained are presented in
Tab. 1 and in Figure 3. The matrix size of the coarse grid problem is 496. The matrix size of the local problem is denoted $S_l$ and $N_H^H$, the number of coarse grid elements in the active region, is 8.

![Graph](image1)

![Graph](image2)

![Graph](image3)

![Graph](image4)

Figure 3: Error of the uniform coarse grid solution in (a). In (b), the local problem solution error and in (c) the updated coarse grid solution. In (d), the coarse grid solution is plotted on the same colour scale as in (a) and (b).

In Tab. 1, using a uniform grid would have resulted into a system of size 1984, 7936 and 31744 respectively for $h = 0.5, 0.25$ and 0.125! In the case of a composite grid solution, the systems would have sizes 520, 616 or 1000 respectively. We note from the table that the algorithm converges in one step which involves solving two global coarse grid size problems and one local problem.

Let $N$ be the size of the global problem and $M$ the size of the local problem. We assume that the local region is only a small part
of the domain such that $N - N^H_a \approx N$, i.e., $N^H_a \ll N$. For example, in our experiments, $N = 496$ and $N^H_a = 8$. The operational count for LU-decomposition is $\frac{1}{3}N^3$ for a size $N$ matrix. After refinement, we have a composite problem of complexity

$$\frac{1}{3}(N + M)^3 \approx \frac{8}{3}N^3$$

if $M \approx N$. \hfill (29)

The BEM-LDC algorithm converges in one step and has total complexity

$$2 \cdot \frac{1}{3}N^3 + \frac{1}{3}M^3 \approx N^3.$$ \hfill (30)

Comparing (29) and (30) clearly shows that the BEM-LDC algorithm is at least twice less complex than solving the composite problem.

With the same coarse grid problem, if we were to refine globally in the same procedure as illustrated in Fig. 2, then achieving a grid of $H = 0.125$ would result in a system of size $128N$ with a complexity of $\frac{1}{3}(128N)^3$. Another pay-off when we use the BEM-LDC algorithm is the lesser need for memory. This is even more vivid when we weigh BEM-LDC over uniform global refinement.

<table>
<thead>
<tr>
<th>H</th>
<th>1.0</th>
<th>1.0</th>
<th>1.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>h</td>
<td>0.500</td>
<td>0.250</td>
<td>0.125</td>
</tr>
<tr>
<td>$|e_0^H|_\infty$</td>
<td>1.416616</td>
<td>1.416616</td>
<td>1.416616</td>
</tr>
<tr>
<td>$|e_1|_\infty$</td>
<td>0.747629</td>
<td>0.395171</td>
<td>0.393202</td>
</tr>
<tr>
<td>$|e_2|_\infty$</td>
<td>0.740818</td>
<td>0.394991</td>
<td>0.392970</td>
</tr>
<tr>
<td>$|e_3|_\infty$</td>
<td>0.740818</td>
<td>0.394989</td>
<td>0.392966</td>
</tr>
<tr>
<td>$S_l$</td>
<td>32</td>
<td>128</td>
<td>512</td>
</tr>
</tbody>
</table>

Table 1: Uniform course grid errors ($\|e_0^H\|_\infty$) and errors after the first, second and third iterations of the BEMLDC Algorithm, $N^H_a = 8$; $S_l$ is the size of the local problem.
Even for a simple model problem on a PC with 512MB, \( H = 0.5 \) is the minimum yet on the same PC we could go up to \( h = 0.125 \) using the the BEM-LDC algorithm.

### 4 Conclusions

In this article we have presented a Local Defect Correction technique for the Boundary Element Method. LDC is an iterative process suitable for efficiently solving problems characterised by a region of high activity that covers a small part of the boundary for the physical domain. Examples are the Impressed Current Cathodic Protection problems such as those discussed in [Kakuba, 2005] and [Lenaers, 2006]. The problem is first solved on a global coarse grid; the computed coarse grid forms part of the boundary condition for a local problem where a more accurate solution can be computed by means of a smaller grid size. The global coarse grid solution and the local fine grid solution are combined in a special way in an iterative manner through defect correction to improve the first coarse grid approximation. The new coarse grid approximation can in turn provide a boundary condition for a new local problem. The process is repeated till convergence, which is in general very fast. Results of numerical experiments show that BEM-LDC can achieve the same accuracy as a uniform grid solver whose grid size coincides with that used in the local problem of BEM-LDC algorithm. But BEM-LDC guarantees a lower computational cost than the uniform grid solver.

### References


