Section Eight

Computational and Mathematical Aspects
Some Localized Boundary–Domain
Integro–Differential Formulations for
Quasi–linear Problems with Variable Coefficients

S.E. Mikhailov
Div. of Mathematics, Glasgow Caledonian University,
Glasgow, G4 0BA, UK, s.mikhailov@gcal.ac.uk

Keywords: Non–linear, Quasi–linear, Partial Differential Equation, Localized, Boundary-Domain Integro-Differential Equation

Abstract. Boundary value problems for a second order nonlinear partial differential equation with variable coefficients depending on the unknown solution and its gradient are considered. Specially constructed localized parametrixes of an auxiliary linear partial differential equation and a combination of the Green identities for the original and auxiliary equations are used in this paper to reduce the nonlinear BVP to a nonlinear Localized Boundary-Domain Integro-Differential Equation (LBDIDE). After discretization, this results in a sparsely populated system of nonlinear algebraic equations, which can be solved by well-known efficient methods. Some techniques of the parametrix localization are discussed and the corresponding nonlinear LBDIDEs are presented.

Introduction

Application of the Boundary Integral Equation (BIE) method (boundary element method, elastic potential method) to solution of linear Boundary Value Problems (BVPs) for Partial Differential Equations (PDEs) has been intensively developed over recent decades. A non-linear BVP can be reduced to a non-linear boundary-domain integral equation, see e.g. [1–3], using a fundamental solution of an auxiliary linear PDE (with coefficients equal ones of the original PDE at zero value of solution). However, the fundamental solution is usually highly non-local, which leads after discretization to a system of nonlinear algebraic equations with a fully populated matrix. Moreover, the fundamental solution is generally not available in an explicit form if the coefficients of the auxiliary PDE vary in space.

To prevent these difficulties, some localized parametrixes were constructed and used in [4] reducing linear BVPs with variable coefficients to linear lo-
calized BDIDEs. The approach was extended in [5] to reduction of BVPs for a second order nonlinear (quasi-linear) elliptic PDE with variable coefficients to quasi-linear localized BDIDEs. That method however would lead to an LBDIDE including the second order derivatives, if the coefficients of the BVP depend not only on the unknown solution but also on its gradient. To obtain an LBDIDE with only first derivatives, a linear combination of the Green identities for the original and an auxiliary PDE is implemented in this paper.

**BVP, Green Identity, Parametrix and Integro–Differential Equations**

For illustration of the general approach we consider a mixed boundary–value problem for the following quasi-linear equation of the second order in a 2D or 3D open domain Ω,

\[
\begin{align*}
[L(x)u](x) &:= \frac{\partial}{\partial x_k} \left[ a(\nabla u(x), u(x), x) \frac{\partial u(x)}{\partial x_k} \right] = f(x), \quad x \in \Omega \\
\begin{cases}
  u(x) = \bar{u}(x), & x \in \partial_D \Omega \\
  \frac{\partial u(x)}{\partial n} = \bar{t}(x), & x \in \partial_N \Omega
\end{cases}
\end{align*}
\]

where \( u(x) \) is unknown, the coefficient \( a(\nabla u, u, x) > C > 0 \) is a known function of the unknown solution \( u(x) \) and its gradient \( \nabla u(x) \); \( f(x) \) is a known right hand side, \( n(x) \) is an outward normal vector to the boundary \( \partial \Omega \), \( \bar{u}(x) \) and \( \bar{t}(x) \) are known functions on the parts \( \partial_D \Omega \) and \( \partial_N \Omega \) of the boundary, respectively. Summation in repeated indices is supposed from 1 to 2 in the 2D and from 1 to 3 in the 3D case unless stated otherwise. Such problems appear particularly in the non-linear elasticity (anti–plane problem) and in the stationary potential flow of a compressible fluid.

Let us fix a point \( y \) and consider a linear differential operator with constant coefficients \( [L(y)v](x) := \frac{\partial}{\partial x_k} \left[ a(\nabla u(y), u(y), y) \frac{\partial v(x)}{\partial x_k} \right] \) and the corresponding boundary flux operator \( [T(y)v](x) := a(\nabla u(y), u(y), y) \frac{\partial v(x)}{\partial n} \). If we write the first Green identities for the operators \( [L(x)u](x) \) and \( [L(y)u](x) \) and subtract them from each other, we obtain a counterpart of the second Green identity,

\[
\begin{align*}
\int_{\Omega} \left\{ u(x)[L(y)v](x) - v(x)[L(x)u](x) \right\} d\Omega(x) &= \\
\int_{\partial_D \Omega} \left\{ u(x)[T(y)v](x) - v(x)[T(x)u](x) \right\} d\Gamma(x) +
\end{align*}
\]
where \( u(x) \) and \( v(x) \) are arbitrary functions for that the operators and integrals in eq (4) have sense.

Let \( P(y)(x, y) \) be a parametrix for the linear differential operator \([L(y)v](x)\) associated with the point \( y \), that is,

\[
[L(y)P(y)(\cdot, y)](x) := \frac{\partial}{\partial x_k} \left[ a(\nabla u(y), u(y), y) \frac{\partial P(y)(x, y)}{\partial x_k} \right] = \delta(x - y) + R(y)(x, y),
\]

where the remainder term \( R(y)(x, y) \) is of the Laplace operator, \( F(y)(x, y) = F_\Delta(x, y)/a(\nabla u(y), u(y), y) \). Denoting \( |x - y| = \sqrt{(x_k - y_k)(x_k - y_k)} \), we have,

\[
F(y)(x, y) = \frac{\ln |x - y|}{2\pi a(\nabla u(y), u(y), y)}, \quad x, y \in \mathbb{R}^2 \tag{5}
\]

\[
F(y)(x, y) = \frac{-1}{4\pi a(\nabla u(y), u(y), y)|x - y|}, \quad x, y \in \mathbb{R}^3. \tag{6}
\]

Assuming \( u(x) \) is a solution of PDE (1) and using a parametrix \( P(y)(x, y) \) as \( v(x) \) in the Green identity (4), one can obtain the following non-linear boundary–domain integro–differential relation,

\[
c(y)u(y) - \int_{\partial \Omega} u(x)[T(y)P(y)(\cdot, y)](x)d\Gamma(x) + \int_{\partial \Omega} P(y)(x, y)[T(x)u](x)d\Gamma(x)
- \int_{\Omega} \frac{\partial P(y)(x, y)}{\partial x_k} [a(\nabla u(x), u(x), x) - a(\nabla u(y), u(y), y)] \frac{\partial u(x)}{\partial x_k} d\Omega(x)
+ \int_{\Omega} R(y)(x, y)u(x)d\Omega(x) = \int_{\Omega} P(y)(x, y)f(x)d\Omega(x), \tag{7}
\]

\[
c(y) := \{ 1 \text{ if } y \in \Omega, \ 0 \text{ if } y \notin \Omega, \ \alpha(y)/(2\pi) \text{ if } y \in \partial \Omega \text{ and } \Omega \subset \mathbb{R}^2, \ 
\alpha(y)/(4\pi) \text{ if } y \in \partial \Omega \text{ and } \Omega \subset \mathbb{R}^3 \},
\]

where \( \alpha(y) \) denotes an interior space angle at a point \( y \) of the boundary \( \partial \Omega \). If the parametrix is a fundamental solution, \( P(y)(x, y) = F(y)(x, y) \), then the last integral disappears in the left hand side of eq (7).
Using boundary conditions (2), (3), one can consider eq (7) as a BDIDE. Note that the BDIDE includes at most the first derivatives of the unknown solution \( u(x) \) both directly in the first domain integral term in the left hand side and through the coefficient \( a(\nabla u, u, \cdot) \) in the operators \( T(x) \), \( T(y) \), and the functions \( P^{(y)}(x, y) \) and \( R^{(y)}(x, y) \). The BDIDE can be reduced after some discretization to a system of nonlinear algebraic equation and solved numerically. The system will include unknowns not only at the boundary but also at internal points. Moreover, since the commonly used parametrixes, e.g. fundamental solutions (5), (6), are highly non-local, the matrix of the system will be fully populated and this makes its numerical solution more expensive. To avoid this difficulty, we present below some ideas of constructing localized parametrixes and consequently localized BDIDEs (LBDIDEs).

Localized Parametrix and BDIDE/BDIDP

A parametrix is not unique and all parametrixes for the differential operator \( L^{(y)} \) have the same singularity at \( x = y \) but can differ at other points. Thus we can consider a function \( P^{(y)}(x, y) = \chi(x, y)P^{(y)}(x, y) \), where \( P^{(y)} \) is an available (not localized) parametrix, e.g. a fundamental solution \( F^{(y)}(x, y) \), and \( \chi(x, y) \) is a cut-off function, such that \( \chi(x, y) = 1 \) and \( \chi(x, y) = 0 \) at \( x \) not belonging to closure of an open localization domain \( \omega(y) \) (a vicinity of \( y \)). Then \( P^{(y)}(x, y) \) has the same singularity as \( F^{(y)}(x, y) \) at \( x = y \) but is localized (non zero) only on \( \omega(y) \). Further we have, \( [L^{(y)}P^{(y)}(\cdot, y)](x) = [L^{(y)}\{\chi(\cdot, y)P^{(y)}(\cdot, y)\}](x) = [L^{(y)}P^{(y)}(\cdot, y)](x) + [L^{(y)}\{(1-\chi(\cdot, y))P^{(y)}(\cdot, y)\}](x) = \delta(x-y) + R^{(y)}(x, y) \), \( R^{(y)}(x, y) = R^{(y)}(x, y) + [L^{(y)}\{(1-\chi(\cdot, y))P^{(y)}(\cdot, y)\}](x) \). Consequently \( R^{(y)} \) will have the necessary properties of the remainder, that is, the localized function \( P^{(y)}(x, y) \) is also a parametrix, at least if \( \chi \) is smooth enough. Note that if \( P^{(y)} \) is a fundamental solution, then \( R^{(y)}(x, y) = 0 \) but generally \( R^{(y)} \neq 0 \).

Suppose \( \chi(x, y) \) is smooth in \( x \in \bar{\omega}(y) \) but not necessarily zero at \( x \in \partial \omega(y) \). Then \( P^{(y)}(x, y) \) is a discontinuous localized parametrix at \( x \in \bar{\Omega} \) and \( P^{(y)}(x, y) = R^{(y)}(x, y) = 0 \) if \( x \notin \bar{\omega}(y) \). Substituting \( P^{(y)}(x, y) \) for \( P^{(y)}(x, y) \) in eq (7) and replacing \( \Omega \) by the intersection \( \omega(y) \cap \Omega \), we arrive at the integral equality localized on \( \omega(y) \cap \Omega \).
constant, \( \chi \)

The last integral in the left hand side of (8) disappears if the localization is a fundamental solution, and we can construct a localized parametrix \( x \in \Omega \) to a Localized Boundary-Domain Integro-Differential Problem for Dirichlet boundary condition (2) at another Localized Boundary-Domain Integro-Differential Problem for Dirichlet boundary condition (2) at use eq (8) only at (8) but leave be employed as a parametrix Green function is available in an analytical form only for sufficiently sim-
P

Similar to [6, 7], the Green function for the operator \( L^{(y)} \) on \( \omega(y) \) may be employed as a parametrix \( P^{(y)}(x, y) \) vanishing on \( \partial \omega(y) \). However, the Green function is available in an analytical form only for sufficiently simple shapes of the localization domain \( \omega(y) \), e.g. for a ball. It seems to be simpler and more universal to construct a proper localized parametrix as \( P^{(y)}(x, y) = \chi(x, y)P^{(y)}(x, y) \), where \( \chi(x, y) \) is a continuous in \( x \in \Omega \) cut-off function, which is smooth in \( \tilde{\omega}(y) \) and equal to zero both on the bound-
ary and outside of $\omega(y)$, whereas $P^{(y)}$ is an available parametrix (e.g. a fundamental solution for $L^{(y)}$).

To simplify the integral representation even further by getting rid of the remaining integral along $\partial\omega(y)$, one can employ a smooth in $x \in \Omega$ cut-off function $\chi(x, y)$, which vanishes on $\partial\omega(y)$ together with its normal derivative in $x$. Then the same holds true also for the parametrix $P^{(y)}_\omega(x, y) = \chi(x, y)P^{(y)}(x, y)$. For such a parametrix, the third and the forth integrals disappear in the left hand side of eq (8). Some examples of internally or globally smooth cut-off functions localized on a ball or on a cube around a point $y$ in $\mathbb{R}^3$, are presented in [4].

It is supposed further to discretize the LBDIDE directly, i.e. not using the integral representations for the gradients and reducing the LBDIDE to a system of non-linear integral equations (Cauchy–singular on the domain and hyper–singular on the boundary) like e.g. in [1–3]. After discretization, the LBDIDE is reduced to some sparsely populated systems of quasi-linear algebraic equations. This can be done e.g. by the collocation method, employing a local interpolation or approximation for the unknown function $u(x)$, which can be associated with a mesh–based or mesh–less discretization similar to [4].

References

Multiwavelet Boundary Integral Solution for Laplace’s Equation

S. Amini\textsuperscript{1} and S. P. Nixon\textsuperscript{2}

\textsuperscript{1}Mathematics, School of Sciences, University of Salford, Salford, M5 4WT, UK. Email: S.Amini@salford.ac.uk
\textsuperscript{2}Mathematics, School of Sciences, University of Salford, Salford, M5 4WT, UK. Email: S.P.Nixon@pgr.salford.ac.uk

Keywords: Multiwavelets, wavelets, Laplace’s equation, boundary integral solution, nonstandard method.

Abstract. In this paper we consider the application of the non-standard representation of an operator [1] and multiwavelets for the solution of boundary integral equations with Calderón-Zygmund type kernels. In particular we consider the boundary integral equation for the Laplace problem in 2 dimensions.

Introduction

In general the solution of boundary integral equations will result in a linear system with full coefficient matrices. Therefore we wish to use a basis that will result in sparse coefficient matrices, such as orthogonal wavelets with compact support and $\mu$ vanishing moments. However, one major drawback is that in order to have $\mu$ vanishing moments, the support of the wavelet, $\psi$, must be at least $[0, \mu]$. This causes difficulties with the practical use of wavelets, particularly at edges and corners. To avoid such problems, in this paper we employ the multiwavelets of [2]. With such basis functions the order of vanishing moments $\mu$ is related to the number of ‘mother’ wavelets rather than the size of the compact support.

A fast algorithm, using multiwavelets and the non-standard representation of an operator [1], is presented requiring $O(n \log n)$ operations for matrix-vector multiplication. A consequence of using the non-standard representation of our operator is that the wavelet levels are separated. This results in a matrix representation of our operator containing only interactions between wavelets and scaling functions at the same level. Upper bounds for the size of elements of the non-standard representation are obtained. This is used in order to carry out a complete error analysis and find a band outside of which elements can be set to zero without affecting the level of discretisation error. Finally we present a number of numerical results to verify the theory.
Multiwavelets on [0,1]

Suppose \( k \) is a positive integer and \( m \) a non-negative integer, we define the space \( V^k_m \) of piecewise polynomial functions \( V^k_M = \{ f : f|_{I_n} = [2^{−m}n, 2^{−m}(n+1)] \} \) is a polynomial of degree less than \( k \) \( \forall n = 0, 1, \ldots, 2^m - 1 \) and vanishes elsewhere. It is clear that \( V^k_0 \subset V^k_1 \subset \ldots \subset V^k_m \subset \ldots \). For \( m = 0, 1, 2, \ldots \), we define the space \( W^k_m \) to be the orthogonal complement of \( V^k_m \) in \( V^k_{m+1} \); that is \( V^k_{m+1} = V^k_m \oplus W^k_m \). Then we have the decomposition \( V^k_M = V^k_0 \oplus V^k_1 \oplus W^k_0 \oplus W^k_1 \oplus \ldots \oplus W^k_{m-1} \).

The space \( V^k_0 \) is the space of polynomials of degree less than \( k \) on the interval \([0,1]\) and we assume \( \{ \phi_1, \phi_2, \ldots, \phi_k \} \) is a basis for it. Suppose \( \{ \psi_1, \psi_2, \ldots, \psi_k \} \) is a basis of \( W^k_0 \). Therefore, for the orthogonality condition \( V^k_0 \perp W^k_0 \) to be satisfied we require the first \( k \) moments of \( \{ \psi_1, \ldots, \psi_k \} \) to vanish. That is \( \int_0^1 \psi_j(x)x^i \, dx = 0 \) for \( j = 1, 2, \ldots, k; \ i = 0, 1, \ldots, k - 1 \).

The \( 2k \)-dimensional space \( W^k_m \) is spanned by the functions \( \{ \psi_1(2x), \ldots, \psi_k(2x), \ \psi_1(2x-1), \ldots, \psi_k(2x-1) \} \) and if we define \( \psi_j^{m,n} = 2^\frac{m}{2} \psi_j(2^m x - n) \) the space \( W^k_m \) is spanned by \( \{ \psi_j^{m,n} \} \) for \( j = 1, 2, \ldots, k \) and \( n = 0, 1, \ldots, 2^m - 1 \). Therefore, if we know a basis for \( W^k_0 \) it is possible to generate all spaces \( W^k_m \). Similarly \( \phi_j^{m,n} = 2^\frac{m}{2} \phi_j(2^m x - n) \) for \( j = 1, 2, \ldots, k \) and \( n = 0, 1, \ldots, 2^m - 1 \) span the space \( V^k_m \).

The multiwavelets are obtained from the mother wavelets \( \psi_j \) which are generated by using the procedure described in [2]. This makes use of the Gram-Schmidt orthogonalisation process. We consider the cases when \( k = 1, 2, 3, 4 \).

Multiwavelet Galerkin Method

The standard form of the Galerkin representation of the operator \((I - K)\) is the matrix \( K = (I - P_M K)P_M \) where \( P_M : L^2[0,1] \rightarrow V^k_M \) is the projection operator. Here we use the non-standard representation of an operator, introduced in [1], as this groups together the different resolutions of spaces, making it possible to use a measure of distance to obtain a bound on the elements of the matrix.

Consider the projection operators \( P_m : L^2[0,1] \rightarrow V^k_m \) and \( Q_m : L^2[0,1] \rightarrow W^k_m \). Given the second kind operator \((I - K) : L^2[0,1] \rightarrow L^2[0,1]\), its discretisation on the space \( V^k_M \) will be \((I - P_M K)P_M \). Following [1] and the fact that \( Q_{m-1} = P_m - P_{m-1} \), we write \((I - P_M K)P_M \) as

\[
(I - P_M K)P_M = \sum_{m=1}^{M} [(I - Q_{m-1} K)Q_{m-1} - Q_{m-1} K P_{m-1} - P_{m-1} K Q_{m-1}] + (I - P_0 K)P_0
\]

(1)
where

\[(I - Q_mK)Q_m f = \sum_{j_1,n_1,j_2,n_2} f^m_{j_1,n_1}(\delta_{j_1,j_2} \cdot \delta_{n_1,n_2} - \alpha^m_{j_1,n_1,j_2,n_2}) \psi^m_{j_2,n_2}(x), \quad (2)\]

\[Q_mKP_m f = \sum_{j_1,n_1,j_2,n_2} f^m_{j_1,n_1}\beta^m_{j_1,n_1,j_2,n_2} \psi^m_{j_2,n_2}(x) \quad \text{and} \quad (3)\]

\[P_mKQ_m f = \sum_{j_1,n_1,j_2,n_2} f^m_{j_1,n_1}\gamma^m_{j_1,n_1,j_2,n_2} \psi^m_{j_2,n_2}(x). \quad (4)\]

Here we have defined \(\alpha^m_{j_1,n_1,j_2,n_2} := \langle K\psi^m_{j_1,n_1}, \psi^m_{j_2,n_2} \rangle\), \(\beta^m_{j_1,n_1,j_2,n_2} := \langle K\phi^m_{j_1,n_1}, \psi^m_{j_2,n_2} \rangle\), and \(\gamma^m_{j_1,n_1,j_2,n_2} := \langle K\phi^m_{j_1,n_1}, \phi^m_{j_2,n_2} \rangle\), where \(\alpha^m\), \(\beta^m\) and \(\gamma^m\) are matrices of order \(k2^m \times k2^m\). Also let \((\chi)_{i,j} := \langle K\phi_i, \phi_j \rangle : i,j = 1, \ldots, k\) be the matrix representation of the operator \(P_0 \mathbb{K}_P\). Then the non-standard form matrix representation of the operator \((I - K)\) will be a matrix, \(K_M\), of order \(2k(2^M - 1)\). Therefore in this method instead of solving \(Kf_h = g_h\), we solve \(R^M K_M D^M f_h = g_h; D^M\) and \(R^M\) are matrices representing the decomposition and reconstruction operators of the multiwavelets; see [3].

**Error Analysis**

To evaluate numerically all the elements of the non-standard matrix is expensive, costing approximately \(cn^2\) operations where the constant \(c\) is large. Therefore we require an a priori bound for the elements of the matrix so that we do not evaluate elements whose sizes are going to be smaller than some prescribed threshold, \(\eta\) say, see [3].

We also need to find an estimate for the error in the solution caused by setting all elements less than \(\eta\) to zero. Let \(K_M^0\) be the matrix \(K_M\) with all elements of size less than \(\eta\) set to zero. Let \(\tilde{K} = R^M K_M D^M\) and \(\bar{K} = R^M K_M^0 D^M\), where \(R^M\) and \(D^M\) are the multiwavelet reconstruction and decomposition matrices. Therefore instead of solving \(\tilde{K} f_h = g_h\), we wish to solve \(\bar{K} f_h = g_h\). Following [4], if \(f_h \in C^r\), \(r \geq k\) and also \(\eta = \frac{h^k}{M}\), then we obtain \(\|f_h - \bar{f}_h\| \leq O(h^k)\), where \(h = 2^{-M}\).

We can bound the elements of \(\alpha^m\), \(\beta^m\) and \(\gamma^m\) by

\[|\alpha^m_{j_1,n_1,j_2,n_2}|, |\beta^m_{j_1,n_1,j_2,n_2}|, |\gamma^m_{j_1,n_1,j_2,n_2}| \leq \frac{c(m)2^\frac{r}{2}}{(|n_1 - n_2| - 1)^{k+1}k!k^{\sqrt{k} + 1}}, \quad (5)\]

see [5], and we can therefore predict the position of the elements of matrix \(K_M\) that are nearly zero. Let us assume that they are outside a band from the diagonal of size \(B_m\) in \(\alpha^m\), \(\beta^m\) and \(\gamma^m\), that is when \(|n_1 - n_2| > B_m\). We then replace the matrix \(K_M\) with \(K_M^B\), a modified version of \(K_M\) where all the elements outside of
a band $B_m$ on level $m$ have been set to zero. Thus we wish to derive criteria for choosing $B_m$ in such a way that $\|K_M - K_M^B\|_\infty \leq \eta$.

Let us consider the submatrices $\alpha^{m-1}, \alpha^{M-2}, \ldots, \alpha^0$, and $\alpha^{M-1}, \alpha^{M-2}, \ldots, \alpha^0$ obtained from $\alpha^m$ by setting to zero all elements $\alpha_{j_1,n_2}^{m-1,j_2,n_2}$ such that $|n_1 - n_2| > B_m$. The following result gives a bound for the difference between $\alpha^m$ and $\alpha^m_B$.

**Lemma 1.** Let $\bar{\alpha}^m = \alpha^m - \alpha^m_B$, $\bar{\beta}^m = \beta^m - \beta^m_B$ and $\bar{\gamma}^m = \gamma^m - \gamma^m_B$ then

$$\|\bar{\alpha}^m\|_\infty, \|\bar{\beta}^m\|_\infty, \|\bar{\gamma}^m\|_\infty \leq \frac{2\pi c(m)(k+1)}{B_m k! k\sqrt{2k+1}}.$$ (6)

If we let $K_M - K_M^B = \begin{pmatrix} \Gamma^{M-1} & 0 \\ \cdot \cdot \cdot \\ 0 & \Gamma^0 \end{pmatrix}$, be a real block diagonal matrix such that all off diagonal blocks are zero and all diagonal blocks $\Gamma^{M-1}, \ldots, \Gamma^0$ are square matrices of size $K^{2m+1} \times k^{2m+1}$ defined like $\Gamma^m = \begin{pmatrix} \bar{\alpha}^m & \bar{\beta}^m \\ \bar{\gamma}^m & 0 \end{pmatrix}$, $m = 0, 1, \ldots, M - 1$. Then

$$\|\Gamma^m\| \leq \frac{2\pi c(m)2\pi (k+1)}{B_m k! k\sqrt{2k+1}}.$$ (7)

Since we require $\|K_M - K_M^B\| \leq \eta$, we also require $\|\Gamma^m\| \leq \eta$ for $m = 0, 1, \ldots, M - 1$. Therefore we choose the size $B_m$ of the band diagonal for each block $\alpha^m, \beta^m$ and $\gamma^m$ to

$$B_m = \left( \frac{2\pi c(m)2\pi (k+1)}{\eta k! k\sqrt{2k+1}} \right) \quad m = 0, 1, \ldots, M - 1.$$ (8)

**Numerical Results**

In this section we present numerical results for the Laplace problem exterior to an ellipse of circumference $4\pi$, with major axis $2.2740969$ and minor axis $1.7055727$ centered at the origin. The problem is equivalent to that generated by a point source placed at $P_0 = (1.1875, 2.5\pi)$ with strength $1$. That is the field generated by $f(P) = -\frac{1}{2\pi} \ln|P - P_0|$. In the tables 1-4, $\|f - f_h^\text{NS}\|$ is the $L_2$ norm of the error using the classical Galerkin method with the scaling functions as the basis. $\|f - f_h^\text{NS}\|$ is the $L_2$ error using the non-standard Galerkin method and $\|f - f_h^{N\text{SO}}\|$ is the $L_2$ error for the non-standard Galerkin method, where the elements outside of a band, $B_m$, have been set to zero.
## Advances in Boundary Element Techniques

### Table 1: K=1

<table>
<thead>
<tr>
<th>M</th>
<th>( |f - f_h| )</th>
<th>( |f - f_h^{NS}| )</th>
<th>( |f - f_h^{NSO}| )</th>
<th>compression %</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>1.5261639 × 10^{-2}</td>
<td>1.5248032 × 10^{-2}</td>
<td>1.4837710 × 10^{-2}</td>
<td>18.8</td>
</tr>
<tr>
<td>4</td>
<td>7.3506907 × 10^{-3}</td>
<td>7.3493441 × 10^{-3}</td>
<td>7.4543544 × 10^{-3}</td>
<td>51.6</td>
</tr>
<tr>
<td>5</td>
<td>3.8222310 × 10^{-3}</td>
<td>3.8220649 × 10^{-3}</td>
<td>4.5156395 × 10^{-3}</td>
<td>73.8</td>
</tr>
<tr>
<td>6</td>
<td>1.9554775 × 10^{-3}</td>
<td>1.9554752 × 10^{-3}</td>
<td>2.4968148 × 10^{-3}</td>
<td>86.8</td>
</tr>
<tr>
<td>7</td>
<td>9.8903115 × 10^{-4}</td>
<td>9.8903060 × 10^{-4}</td>
<td>1.3642175 × 10^{-3}</td>
<td>92.4</td>
</tr>
<tr>
<td>8</td>
<td>4.9732308 × 10^{-4}</td>
<td>4.9732348 × 10^{-4}</td>
<td>6.2809880 × 10^{-4}</td>
<td>95.4</td>
</tr>
</tbody>
</table>

### Table 2: K=2

<table>
<thead>
<tr>
<th>M</th>
<th>( |f - f_h| )</th>
<th>( |f - f_h^{NS}| )</th>
<th>( |f - f_h^{NSO}| )</th>
<th>compression %</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>1.8453789 × 10^{-3}</td>
<td>1.8453388 × 10^{-3}</td>
<td>1.8344719 × 10^{-3}</td>
<td>18.8</td>
</tr>
<tr>
<td>4</td>
<td>5.6171957 × 10^{-4}</td>
<td>5.6168664 × 10^{-4}</td>
<td>6.2687853 × 10^{-4}</td>
<td>51.6</td>
</tr>
<tr>
<td>5</td>
<td>1.3916301 × 10^{-4}</td>
<td>1.3915999 × 10^{-4}</td>
<td>1.4944720 × 10^{-4}</td>
<td>63.3</td>
</tr>
<tr>
<td>6</td>
<td>3.4385048 × 10^{-5}</td>
<td>3.4384062 × 10^{-5}</td>
<td>3.6389729 × 10^{-5}</td>
<td>69.1</td>
</tr>
<tr>
<td>7</td>
<td>8.5346164 × 10^{-6}</td>
<td>8.5345794 × 10^{-6}</td>
<td>1.1220857 × 10^{-5}</td>
<td>75.6</td>
</tr>
<tr>
<td>8</td>
<td>2.1254632 × 10^{-6}</td>
<td>2.1254430 × 10^{-6}</td>
<td>2.2171610 × 10^{-6}</td>
<td>73.5</td>
</tr>
</tbody>
</table>

### Table 3: K=3

<table>
<thead>
<tr>
<th>M</th>
<th>( |f - f_h| )</th>
<th>( |f - f_h^{NS}| )</th>
<th>( |f - f_h^{NSO}| )</th>
<th>compression %</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>3.6276295 × 10^{-4}</td>
<td>3.6278469 × 10^{-4}</td>
<td>3.5345361 × 10^{-4}</td>
<td>18.8</td>
</tr>
<tr>
<td>4</td>
<td>4.0682455 × 10^{-5}</td>
<td>4.0682193 × 10^{-5}</td>
<td>4.2924098 × 10^{-5}</td>
<td>46.9</td>
</tr>
<tr>
<td>5</td>
<td>5.4542842 × 10^{-6}</td>
<td>5.4547906 × 10^{-6}</td>
<td>5.6395228 × 10^{-6}</td>
<td>60.6</td>
</tr>
<tr>
<td>6</td>
<td>6.8821017 × 10^{-7}</td>
<td>6.8830250 × 10^{-7}</td>
<td>7.0708284 × 10^{-7}</td>
<td>63.3</td>
</tr>
<tr>
<td>7</td>
<td>8.6220004 × 10^{-8}</td>
<td>8.6234076 × 10^{-8}</td>
<td>8.7846760 × 10^{-8}</td>
<td>69.1</td>
</tr>
<tr>
<td>8</td>
<td>1.0785156 × 10^{-8}</td>
<td>1.0787199 × 10^{-8}</td>
<td>1.0968565 × 10^{-8}</td>
<td>72.1</td>
</tr>
</tbody>
</table>

### Table 4: K=4

<table>
<thead>
<tr>
<th>M</th>
<th>( |f - f_h| )</th>
<th>( |f - f_h^{NS}| )</th>
<th>( |f - f_h^{NSO}| )</th>
<th>compression %</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>3.6927529 × 10^{-5}</td>
<td>3.6918974 × 10^{-5}</td>
<td>3.9249977 × 10^{-5}</td>
<td>18.8</td>
</tr>
<tr>
<td>4</td>
<td>3.9212541 × 10^{-6}</td>
<td>3.9210285 × 10^{-6}</td>
<td>3.9687877 × 10^{-6}</td>
<td>28.1</td>
</tr>
<tr>
<td>5</td>
<td>2.2001867 × 10^{-7}</td>
<td>2.2013171 × 10^{-7}</td>
<td>2.9656660 × 10^{-7}</td>
<td>63.7</td>
</tr>
<tr>
<td>6</td>
<td>1.3678638 × 10^{-8}</td>
<td>1.3672609 × 10^{-8}</td>
<td>1.4065833 × 10^{-8}</td>
<td>63.3</td>
</tr>
<tr>
<td>7</td>
<td>8.5276574 × 10^{-10}</td>
<td>8.5642710 × 10^{-10}</td>
<td>1.0402139 × 10^{-9}</td>
<td>69.7</td>
</tr>
<tr>
<td>8</td>
<td>5.3224104 × 10^{-11}</td>
<td>5.3562258 × 10^{-11}</td>
<td>5.4166064 × 10^{-11}</td>
<td>72.1</td>
</tr>
</tbody>
</table>
For each of the cases we observe that the error behaves as \(O(h^K)\), as expected by the theory. The column with the errors \(\|f - f_{NS}^h\|\) shows that the non-standard scheme gives the same level of accuracy as the classical piecewise polynomial spaces. The column with the errors \(\|f - f_{NS}^h\|\) shows that we can set to zero those elements of the non-standard matrix which sufficiently small without adversely affecting the level of accuracy. The last column gives the percentage of the elements which have been set to zero in obtaining the solution \(f_{NS}^{h,0}\).

**Conclusion**

For the boundary integral equation for the Laplace problem we have presented conditions that allows us to know a priori which elements are going to smaller than our desired tolerance, so that we can avoid computing them. An \(O(N \ln N)\) algorithm is obtained for the Laplace problem and results given.

Further work includes investigating more complex scenarios including, closed contour shapes with sharp curvature or corners, also investigating the use of a tighter bound on the \(a_m\) matrices.

**References**


Application of Computing Point Method in BEM to the Problem with Biharmonic Differential Operator

N. Kamiya and T. Hashizume
Graduate School of Information Science, Nagoya University, Nagoya 464-8601, Japan

Key Words: Boundary Element Method, Computing Point Method, Biharmonic Equation

Abstract  In this paper, an application of the computing point method, proposed by the author for boundary only computation of nonlinear and/or inhomogeneous problem by the boundary element method, is presented to the problem with biharmonic differential operator. The original problem is decomposed into a system of simultaneous differential equations with similar harmonic differential operator and then boundary element formulation is employed for both equations. Nonlinear and/or inhomogeneous terms multiplied by the fundamental solution to the Laplace equation are converted to the corresponding boundary integrals. Solution procedure and example computation are presented for two-dimensional problems.

Introduction

Recent studies on boundary only computation schemes for nonlinear and/or inhomogeneous problems have developed markedly; that is, Dual Reciprocity Method (DRM) and Multiple Reciprocity Method (MRM) are typical fruits [1, 2]. These are one of the pronounced advantages of the Boundary Element Method (BEM) over other numerical methods known as domain-types such as Finite Difference Method (FDM) and Finite Element Method (FEM). Related researches have also progressed recently such as properties of radial base functions for approximation of nonlinear and/or inhomogeneous terms. Nevertheless appropriate selection of the radial base function and of placement of internal points remain in question.

As an alternative along the similar objective, Kamiya and Xu proposed a method called Computing Point Method (CPM) [3 - 5]. Nonlinear and/or inhomogeneous term is approximated by lower order polynomial in terms of global coordinates and coefficients in it is determined by least squares, which separate solution of discretized entire simultaneous equation from determination of unknown coefficients. In this paper, we apply the computing point method to the problem with biharmonic differential operator as an extension of the original scheme.

Computing Point Method for Nonlinear/Inhomogeneous Problem with Biharmonic Operator

A detailed description of the computing point method is found in References [3 - 5]
with some simple examples. Essential point of the method is that a nonlinear or inhomogeneous term is approximately represented by a polynomial in terms of the global coordinates and their related term appearing in a domain integral is transformed into a boundary integral; that is boundary only formulation of the integral equation. Therefore, boundary only discretization is sufficient. In this scheme, unknown coefficients in the polynomial are determined by least squares for a set points on the boundary as well as inside domain, which are called the computing points. It will, in general, be possible to apply the method to the problem with biharmonic operator using the known fundamental solution and its higher order ones. However, the integral equation formulation for the biharmonic problem requires an additional equation for the boundary derivative besides the equation of the unknown function due to the fourth order differential equation. In this case, singularity in the highest order related fundamental solution is inevitable as well known in elastic thin plate bending problems.

In what follows, we recognize the biharmonic problem as the simultaneous differential equation system of the second order, i.e., Poisson equations. The computing point method is applied to the latter two equations, which will be easier and a sole application of the established scheme.

**Formulation of Integral Equation Formulation**

Consider the following biharmonic differential equation with a nonlinear/inhomogeneous term in a two-dimensional closed domain:

\[ \nabla^4 u(x, y) + \Phi(x, y) = 0 \]  \hspace{1cm} (1)

The inhomogeneous term can be either a known function of the global coordinates or a function containing the unknown function. Now setting another unknown function \( v \) as follows:

\[ \nabla^2 u(x, y) = v \]  \hspace{1cm} (2)

Eq. (1) is represented by the following simultaneous differential equation of the Poisson type:

\[ \nabla^2 u(x, y) = v(x, y) \]  \hspace{1cm} (3)

\[ \nabla^2 v(x, y) + \Phi(x, y, u) = 0 \]  \hspace{1cm} (4)

The right-hand side of Eq. (3) and the inhomogeneous term in Eq. (4) are approximated by polynomials, respectively

\[ v(x, y) = \sum_{j=1}^{15} c_j r_j \]  \hspace{1cm} (5)

\[ \Phi(x, y, u) = \sum_{j=1}^{15} c_j u r_j \]  \hspace{1cm} (6)
where $r_j$ represents terms of the complete polynomials up to the fourth order, i.e., $x, y, x^2, y^2$. Equations (3) and (4) are converted to the integral equations using the fundamental solution of the Laplace equation. Domain integrals corresponding to $v$ in Eq. (3) and $b$ in Eq. (4) appearing in these formulation are transformed into boundary integrals using the polynomial approximations Eqs. (5) and (6). The derived integral equations are discretized by piecewise constant boundary elements. Taking $n$ boundary source points and $m$ internal source points, we obtain the following system of algebraic equations:

$$H_u = G_q + Dc_u$$

$$H_v = G_p + Dc_v$$

$$2\pi u' + H' u = G' q + D' c_u$$

$$2\pi v' + H' v = G' p + D' c_v$$

where $q,p$ are the normal derivatives of $u,v$ on the boundary, $u, q, u', v, p, v'$ are the vectors of $u, q, u', v, p, v'$, and a prime denotes the values inside domain. $c_u, c_v$ are the vectors of $c_u, c_v$. Capital letters $H, G, D, H', G', D'$ are respective coefficient matrices.

The boundary unknown $v$ and the internal unknown $v'$ are able to be represented implicitly by $c_v$ using Eqs. (8), (10) and boundary condition for $v$. The unknown coefficient $c_v$ can be determined by least squares on the selected points, i.e., computing points and consequently, the above-mentioned values are determined. Provided that $v$ is obtained, the right-hand side of Eq. (3) is approximated as Eq. (5); that is, similar least squares are applied to decide the unknown coefficients $c_v$. Thus, the unknown $u$ in Eq. (3) can be determined along similar manner.

**Analysis Process**

Process for obtaining solution by the presented method is as follows:
1) Assume the unknown $u$ (boundary) and $u'$ (inside).
2) Compute $c_u$ by the least squares on the computing points.
3) Solve $v$ and $p$ by Eqs. (8) and (10).
4) Determine $c$, using these values, like process 2).
5) Solve $u$ and $f$ from Eqs. (7) and (9).
6) Compare the just obtained value and the assumed value of $u$. When the difference is sufficiently small, computation terminates. Otherwise, back to process 2) and repeat iteration.

If the inhomogeneous term does not contain the unknown function, the assumption 1) and iterative computation are not needed.

**Examples and Discussion**

We consider here the following two problems in the square domain shown in Fig. 1, i.e., inhomogeneous terms are given as
1) \( b(x, y) = \sin \pi x \sin \pi y \)

2) \( b(x, y, u) = ku(x, y) + \sin \pi x \sin \pi y \quad (k: \text{const.}) \)

The first case is for the known inhomogeneous term and the second is unknown inhomogeneous term because the unknown function is included.

The boundary conditions are thought as \( u = 0, \quad v = 0 \) along the entire boundary, for which analytical solutions are easily obtained as follows:

1) \( u = -\frac{1}{4\pi^2} \sin \pi x \sin \pi y \)

2) \( u = -\frac{1}{4\pi^2 + k} \sin \pi x \sin \pi y \)

Boundary discretization is performed by constant boundary elements of straight line, 8 and 16 elements on each side (total numbers are \( n = 32, \quad 64 \)). 16 computing points on the boundary (BCP) are taken in equal distance (\( \pi = 16 \)). The internal computing points are arranged inside domain as shown in Fig. 1 (\( m=5, \quad 9 \)). Computation are carried out for the combination of the numbers of the boundary element and internal computing points:

\[ (n, m) = (32, 5), (64, 5), (32, 9), \text{and} (64, 9) \]

Computed results of the first problem are compared with the analytical ones in Tables 1 for \( u' \), 2 for \( v' \), respectively. Error in these tables denotes a relative error from the analytical one. Fairly well results are obtained by the present computation scheme. From these results we can notice that the more internal computing points the more accurate the solution.

For the second problem, iteration from an appropriately assumed initial value is required, in this case \( u = u' = 0 \) are employed as the convergence criterion of \( u', 10^{-5} \) \( (n=64, \quad m=9) \). The constant \( k = -1 \) is used. After only three iterations the results were obtained as shown in Table 3. Similarly to the previous problem, very accurate results were found.

Fig. 1 Considered domain and Internal computing points.
Conclusion

The computing point method for the boundary only computation of nonlinear and/or inhomogeneous problem was applied successfully to the problem with biharmonic differential operator. Inhomogeneous terms were thought to be either known function or the case including unknown function. For the latter case iteration scheme was employed to reach the convergence. Further extension and applications will be considered in the future study.

References

5. S.Q. Xu and N. Kamiya, A formulation and solution for boundary element analysis of inhomogeneous-nonlinear problem; The case derivatives of unknown functions are included, *Engineering Analysis with Boundary Elements*, 23, 391-397

Table 1 Results of \( u' \) for the first problem.

<table>
<thead>
<tr>
<th>( n = 32 ) m = 5</th>
<th>Relative Error</th>
<th>( n = 32 ) m = 9</th>
<th>Relative Error</th>
<th>( n = 64 ) m = 5</th>
<th>Relative Error</th>
<th>( n = 64 ) m = 9</th>
<th>Relative Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>-2.61e-03</td>
<td>-2.0e-02</td>
<td>6.9e-03</td>
<td>2.61e-03</td>
<td>-1.7e-02</td>
<td>-2.53e-03</td>
<td>1.0e-02</td>
</tr>
<tr>
<td>f</td>
<td>-1.50e-03</td>
<td>-2.3e-05</td>
<td>-1.50e-03</td>
<td>2.5e-03</td>
<td>-1.50e-03</td>
<td>2.5e-03</td>
<td></td>
</tr>
<tr>
<td>g</td>
<td>-1.50e-03</td>
<td>-2.3e-05</td>
<td>-1.50e-03</td>
<td>2.5e-03</td>
<td>-1.50e-03</td>
<td>2.5e-03</td>
<td></td>
</tr>
<tr>
<td>h</td>
<td>-1.50e-03</td>
<td>-2.3e-05</td>
<td>-1.50e-03</td>
<td>2.5e-03</td>
<td>-1.50e-03</td>
<td>2.5e-03</td>
<td></td>
</tr>
<tr>
<td>i</td>
<td>-1.50e-03</td>
<td>-2.3e-05</td>
<td>-1.50e-03</td>
<td>2.5e-03</td>
<td>-1.50e-03</td>
<td>2.5e-03</td>
<td></td>
</tr>
</tbody>
</table>

Table 2 Results of \( \nu' \) for the first problem.

<table>
<thead>
<tr>
<th>( n = 32 ) m = 5</th>
<th>Relative Error</th>
<th>( n = 32 ) m = 9</th>
<th>Relative Error</th>
<th>( n = 64 ) m = 5</th>
<th>Relative Error</th>
<th>( n = 64 ) m = 9</th>
<th>Relative Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>5.12e-02</td>
<td>-9.6e-03</td>
<td>5.04e-02</td>
<td>5.2e-03</td>
<td>5.11e-02</td>
<td>-8.1e-03</td>
<td>7.6e-03</td>
</tr>
<tr>
<td>b</td>
<td>1.78e-02</td>
<td>-1.4e-02</td>
<td>1.75e-02</td>
<td>-4.1e-05</td>
<td>1.78e-02</td>
<td>-1.5e-02</td>
<td>1.75e-02</td>
</tr>
<tr>
<td>c</td>
<td>1.78e-02</td>
<td>-1.4e-02</td>
<td>1.75e-02</td>
<td>-4.1e-05</td>
<td>1.78e-02</td>
<td>-1.5e-02</td>
<td>1.75e-02</td>
</tr>
<tr>
<td>d</td>
<td>1.78e-02</td>
<td>-1.4e-02</td>
<td>1.75e-02</td>
<td>-4.3e-05</td>
<td>1.78e-02</td>
<td>-1.5e-02</td>
<td>1.75e-02</td>
</tr>
<tr>
<td>e</td>
<td>1.78e-02</td>
<td>-1.4e-02</td>
<td>1.75e-02</td>
<td>-4.0e-05</td>
<td>1.78e-02</td>
<td>-1.5e-02</td>
<td>1.75e-02</td>
</tr>
<tr>
<td>f</td>
<td>2.98e-02</td>
<td>-1.3e-03</td>
<td>2.98e-02</td>
<td>-1.3e-03</td>
<td>2.98e-02</td>
<td>-1.3e-03</td>
<td>2.98e-02</td>
</tr>
<tr>
<td>g</td>
<td>2.98e-02</td>
<td>-1.3e-03</td>
<td>2.98e-02</td>
<td>-1.3e-03</td>
<td>2.98e-02</td>
<td>-1.3e-03</td>
<td>2.98e-02</td>
</tr>
<tr>
<td>h</td>
<td>2.98e-02</td>
<td>-1.3e-03</td>
<td>2.98e-02</td>
<td>-1.3e-03</td>
<td>2.98e-02</td>
<td>-1.3e-03</td>
<td>2.98e-02</td>
</tr>
<tr>
<td>i</td>
<td>2.98e-02</td>
<td>-1.3e-03</td>
<td>2.98e-02</td>
<td>-1.3e-03</td>
<td>2.98e-02</td>
<td>-1.3e-03</td>
<td>2.98e-02</td>
</tr>
</tbody>
</table>
Table 3 Results of $u'$ and $v'$ for the second problem.

<table>
<thead>
<tr>
<th></th>
<th>$u'$</th>
<th>Relative Error</th>
<th>$v'$</th>
<th>Relative Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>-2.53e-03</td>
<td>1.0e-02</td>
<td>5.02e-02</td>
<td>7.6e-03</td>
</tr>
<tr>
<td>b</td>
<td>-8.84e-04</td>
<td>1.9e-03</td>
<td>1.74e-02</td>
<td>6.1e-05</td>
</tr>
<tr>
<td>c</td>
<td>-8.84e-04</td>
<td>1.9e-03</td>
<td>1.74e-02</td>
<td>8.7e-05</td>
</tr>
<tr>
<td>d</td>
<td>-8.84e-04</td>
<td>1.9e-03</td>
<td>1.74e-02</td>
<td>6.8e-05</td>
</tr>
<tr>
<td>e</td>
<td>-8.84e-04</td>
<td>1.9e-03</td>
<td>1.74e-02</td>
<td>5.1e-05</td>
</tr>
<tr>
<td>f</td>
<td>-1.50e-03</td>
<td>2.5e-03</td>
<td>2.97e-02</td>
<td>-1.8e-04</td>
</tr>
<tr>
<td>g</td>
<td>-1.50e-03</td>
<td>2.5e-03</td>
<td>2.97e-02</td>
<td>-1.8e-04</td>
</tr>
<tr>
<td>h</td>
<td>-1.50e-03</td>
<td>2.5e-03</td>
<td>2.97e-02</td>
<td>-1.9e-04</td>
</tr>
<tr>
<td>i</td>
<td>-1.50e-03</td>
<td>2.5e-03</td>
<td>2.97e-02</td>
<td>-1.6e-04</td>
</tr>
</tbody>
</table>

Appendix: Flow of Computation

START

Assume $u$ and $u'$

Compute $c_b$

Solution of $v$ and $p$

Compute $c_v$

Solution of $u$ and $q$

Convergence check of $u$

STOP
A Fast Solution Technique for Large BEM Systems of Equations Using Block Wavelet Transforms

Henrique F. Bucher¹, Luiz C. Wrobel², Webe J. Mansur³ and Carlos Magluta³

¹Center For Computational Research, State University of New York at Buffalo, 9 Norton Hall, UB North Campus, Buffalo, NY, 14260, USA (henrique@bucher.com)

²Department of Mechanical Engineering, Brunel University, Uxbridge UB8 3PH, Middlesex, UK (luiz.wrobel@brunel.ac.uk)

³Department of Civil Engineering, COPPE/Federal University of Rio de Janeiro, PO Box 68506, CEP 21945-910, Rio de Janeiro, Brazil (webe@coc.ufrj.br; magluta@labest.coc.ufrj.br)

Keywords: Boundary elements, wavelet transforms, fast solvers

Abstract. This paper follows an earlier work by Bucher et al. [1] on the application of wavelet transforms to the boundary element method, which shows how to reuse models stored in compressed form to solve new models with the same geometry but arbitrary load cases. The extension presented in this paper involves a new computational procedure created to perform the required two-dimensional wavelet transforms by blocks, theoretically allowing the compression of matrices of arbitrary size in personal computers. A numerical application shows a standard PC being used to solve a 131,072 DOF model, previously compressed, for 100 distinct load cases in less than 1 hour – or 33 seconds for each load case.

Introduction

The solution of very large problems arising from the Boundary Element Method (BEM) is still a challenge for most industrial application developers due to the $O(N^2)$ growth of the system matrices as the number of degrees of freedom in the model increases. Several methods were suggested [2-4] where this explosive growth is substituted by a less restrictive behaviour of order $O(N\log N)$. However, these methods require a complete rewriting of all routines used to perform the calculation of coefficients of the BEM matrices and, most often, also require the development of new companion solvers, upper bounds and respective spectral estimates.
This work proposes a new methodology that extends achievements obtained in a previous work [1] to medium and large problems – of order $10^5$ – while, at the same time, retaining all previous algorithms found in many textbooks, whose characteristics are well studied and documented. The key to this methodology is a block transform that acts as a black box, which can easily be included in standard BEM codes. Furthermore, the use of the Virtual Assembly (VA) technique empowers the block transform such that processed matrices can be reused in their compressed form to rapidly solve an arbitrary number of different load cases.

Review of the Virtual Assembly Technique

The BEM, in its direct formulation for potential problems, generates the following system of equations when the standard point collocation technique is used [5]:

$$Hu = Gq$$  

(1)

According to the VA technique presented by Bucher et al. [1], the original system of equations (1) may be expressed as a function of the compressed boundary element matrices

$$\tilde{H} = \text{threshold}(WHW^T)$$  

(2)

and

$$\tilde{G} = \text{threshold}(WGW^T)$$  

(3)

such that an alternative form of the original system of equations (1) is generated:

$$\left((W^T\tilde{H}W)X_H - (W^T\tilde{G}W)X_G\right)x = f$$  

(4)

where $W$ is the linear operator representing the orthogonal transform used to compress the original matrices $H$ and $G$, very often the wavelet transform. $X_G$ and $X_H$ are diagonal operator matrices with coefficients:

$$X_H_{i,j} = \begin{cases} \lambda_{ij}^H & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$  

and

$$X_G_{i,j} = \begin{cases} \lambda_{ij}^G & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$  

(5)

$\lambda^G$ and $\lambda^H$ are operators responsible for applying boundary conditions and consequently generating the traditional system of equations $Ax=b$. Instead of exchanging rows and columns in the traditional procedure, we write each nodal
value $u$ and its derivative $q$ as a function of the respective unknown $x$ in the solution vector as

$$u = \lambda^H x + u_0$$
$$q = \lambda^G x + q_0$$

(6)

This operation allows solving the system using the two matrices, $G$ and $H$, in their compressed form without generating the system matrix $A$, therefore justifying the expression “virtual assembly”. This technique produces a greater freedom for application of diverse boundary conditions, as shown in [1].

Therefore, the new system matrix is recognized as

$$A = (W^T \bar{H} \bar{W}) X_H - (W^T \bar{G} \bar{W}) X_G$$

(7)

and the force vector is calculated by

$$f = -(W^T \bar{H} \bar{W}) u_0 + (W^T \bar{G} \bar{W}) q_0$$

(8)

As this system of equations remains in the original space, all algebraic properties such as convergence rates and condition numbers are preserved except for the often-negligible distortion introduced by the thresholding operation.

The Block Wavelet Transform

Obtaining the compressed matrices (2) and (3) requires full integration of matrices $H$ and $G$ before the transform and threshold operations can be applied. As the size of the matrices increases quadratically with the number of degrees of freedom, memory consumption becomes a major concern. An effective solution to deal with problems of arbitrary size, independently of the amount of memory available in the computer, is possible by using a novel algorithm developed in this paper, the Block Wavelet Transform (BWT), which allows the compression and reutilization of the compressed blocks in computers equipped with virtually any amount of memory. By partitioning the BEM matrices by blocks, memory requirements are exchanged by time requirements which are very often less restrictive. Blocks can then be freely resized to fit the maximum memory of each computer, therefore maximizing the use of available resources.

The block compression methodology must include a strategy to use the blocks for solving the system in their compressed form, which is done by using a mapping matrix known as adjacency matrix in the finite element terminology [6]. The adjacency matrix allows the description of the original system matrix.
in a series where the coefficients are blocks and the basis functions are the adjacency matrices.

However, not all possible matrix partitions are valid. In other words, there must be a condition that guarantees that no block overlapping or lack of coverage will occur. This work advances beyond the basic adjacency matrix concept by defining a way of restraining the grid partition through an admissibility condition.

**Partition admissibility condition.** A block partition is valid if and only if any matrix $A$ can be reconstructed from its respective blocks $A_k$, $k=1$ to $M$, through the series

$$A = \sum_{k=1}^{M} L_{N,n_k}^{row_k} A_k \left( L_{N,n_k}^{col_k} \right)$$

(9)

The special adjacency matrices $L$ that appear in this series are called **Insertion-Extraction** (IE) matrices to emphasize that these are linked to new matrices with special partitioning properties.

Each IE matrix is denoted by the symbol $L_{N,n}^i$, where $i$ is the index in the expansion series, $N$ is the size of the original, full matrix (assumed to be square), and $n$ is the size of the respective block. All elements in an IE matrix are zero but for those in the diagonal that starts at column $i$, which are ones. This shape provides two important block properties, also illustrated in Figure 1:

**Extraction property.** A square block $A_k$ of size $n$, located at the position $(row_k, col_k)$ of a generic matrix $A$, can be extracted by pre-multiplying the transposed version of the IE matrix $L_{N,n}^{row_k}$ followed by post-multiplying the result by the IE matrix $L_{N,n}^{col_k}$ such that

$$A_k = \left( L_{N,n}^{row_k} \right)^T A L_{N,n}^{col_k}$$

(10)

**Insertion property.** The “$k$” block extracted using the above extraction property can be restored to its original position $(row_k, col_k)$ in the original, full matrix $A$ by pre-multiplying it by the IE matrix $L_{N,n}^{row_k}$ followed by post-multiplying the resulting matrix by the transposed version of the IE matrix $L_{N,n}^{col_k}$.
\[ A_k^{N \times N} = L_{N,i}^{row_k} \cdot A_k \left( L_{N,i}^{col_k} \right)^T \]  

(11)

Both properties can be better understood with the help of Figure 1.

In this way, using the extraction property (10) for decomposition and the insertion property (11) for reconstruction, the square matrix \( A \) can now be sliced into \( M \) distinct blocks \( A_k \), \( k = 1 \) to \( M \), each of them located in one particular position (\( row_k, col_k \)) and with size \( n_k \).

**Application**

To demonstrate the applicability of the present theory, the problem of a concrete column subject to a heat flux – dependent on the exterior temperature – in each side is analysed, as shown in [1]. At the boundaries, the heat flux \( q \) is proportional to the temperature difference between the column \( u \) and the ambient temperature \( u_s \). This boundary condition is expressed by

\[ q = -h(u - u_s) \]  

(12)

where \( h \) is the heat transfer coefficient between the column and the air.

The interior ambient has a temperature of 25°C and, in this region, the column received a surface treatment to lower its heat transfer coefficient to 0.1 cal/m²/s°C. The external ambient has a constant temperature of 0°C and its...
surface is subject to strong convection, which brings its heat transfer coefficient to 0.5 cal/m$^2$/s/°C. The value of the heat transfer coefficient decreases linearly to zero in the central region.

A BEM model was created using 32,768 constant elements per side, which totals 131,072 degrees of freedom. A Pentium III 900MHz with 1.5GB RAM was used to solve this problem. The wavelet Daubechies with 6 coefficients [7] was used for the necessary transform.

In the assembly phase, the $G$ and $H$ matrices were partitioned into 256 blocks of identical size – 8,192. Blocks were calculated, compressed and then saved in XDR format. This process generated 256 files that consumed 360 Megabytes of disk space (roughly a single CD), against 262 Gigabytes that would be necessary to store the two uncompressed matrices, which results in a final compression ratio of 720 to 1.

In the solution phase, combining 10 different external heat transfer coefficients and 10 internal heat transfer coefficients created a set of 100 different load cases. Each load case was solved using the same parameters: GMRES iterative solver with diagonal preconditioning; stopping criterion: relative residual RMS error in the force vector less than $10^{-12}$; and restart in 25 iterations. The diagonal preconditioning feature was only possible because the diagonal elements were stored uncompressed within the compressed blocks where they originally lie.

As these problems were impossible to run with standard iterative solvers (matrix in uncompressed form) or Gauss elimination, we estimated the error by interpolating the value of a 8,192 DOF problem solved with a Gauss solver into the 131,072 elements of the current one, which is not a bad estimate since most of the temperature and heat fluxes variations are smooth. The relative RMS error of the temperature was not allowed to grow above $10^{-3}$.

The loading time of the whole compressed structure was, on average, 6.2 seconds, which is roughly the time it takes to evaluate the integrals of a single 8,192-element block. This low time is mostly due to the adopted XDR binary format since a text format would take at least 6 minutes and 40 seconds to load, according to our previous estimates. Using XDR not only retained the loading speed at low values but also made it possible to use this dataset in several different platforms.

After loading, no significant operation takes place until the GMRES cycles, each of which took, on average, 1.7 seconds to complete. The number of iterations per load case, for the same tolerance value of $10^{-12}$, was roughly the same for most load cases, with a minimum of 8 and a maximum of 40
iterations. The total wall clock time spent to calculate all 100 load cases was 56 minutes and 21 seconds.

**Conclusions**

The results of this work indicate that there is a large class of industrial problems that can greatly benefit from the BWT-VA approach, particularly large models with multiple load cases. These problems benefit from the reusability of the compressed matrices provided by the current approach.

The numerical application demonstrated that a 131,072 DOF problem, which would otherwise require 262 Gigabytes of disk storage, can be compressed to fit the size of a single CD and then reused at any time to solve many different load cases with no need for decompression. Very high compression ratios were obtained (720 to 1), making it possible to achieve very fast loading (6.2 seconds) and solution (33.1 seconds) steps for a 131 thousand DOF problem, which was solved in a rather standard desktop computer.

**References**


COARSE PRECONDITIONING FOR MULTI-DOMAIN ELASTOSTATIC PROBLEMS WITH VARIABLE CONTACT

M. T. ALONSO RASGADO and K. DAVEY
Department of Mechanical, Aerospace and Manufacturing Engineering, UMIST, Manchester.

SUMMARY

Multi-domain elastostatic problems can often be efficiently solved using coupled single-domain iterative techniques. A particular difficulty however is the decrease in convergence rate associated with the increase in number of solution domains. Further convergence difficulties are encountered for multi-domain problems where contacting domains suffer variable contact conditions. This paper is concerned with the application and development of a coarse preconditioner designed to enhance the overall system stability for problems of this type. The governing equations across domains are coupled by means of a multiplicative-Schwarz method for non-overlapping domains as focus is on the use of serial processing. The coarse preconditioner is obtained from a crude representation of the global system of equations although high accuracy is generally required at the contacting interfaces. Attention is restricted to thermo-elastostatic problems with domains connected through spring interfaces. The effect of lowering and increasing the interfacial stiffness between domains is investigated. Computation times are determined for the iterative procedures and for elimination techniques indicating the relative benefits for problems of this nature.

KEY WORDS: casting; variable contact; boundary elements; elastostatics.

1. INTRODUCTION

This paper is concerned with the development of a coarse preconditioner for multi-domain problems that are subject to thermal loading prior and subsequent to assembly. Problems of this type arise in pressure die casting, where die casting dies are typically formed by the assemblage of die blocks and inserts. The dies are required to be of high precision and individual blocks are required to be parted and brought together repeatedly during the casting process. Attention is restricted to multi-domain as opposed to sub-domain as these give rise to contact problems. Only limited work has been done to-date on coarse preconditioning of boundary element system assemblies. Consider the systems of boundary element equations that arise from the discretisation of boundary integral formulation of the governing differential equations for thermoelastic problems, i.e.
where \( u_{\alpha}^* \) and \( p_{\alpha}^* \) are the fundamental solutions for displacement and traction, the remaining variables take their usual meaning. Note, \( N_q \) are shape functions, \( E_i \) is the number of elements on \( \Gamma^i \) and \( N_e \) is the number of nodes on an element.

Consider \( N \) domains and let \( \Gamma^i = \Gamma^i_u \cup \bigcup_{j=1}^{N_e} \Gamma^i_j \) where \( \Gamma^i_j \) is the part of \( \Gamma^i \) that is identified with domains \( i \) and \( j \), and \( \Gamma^i_o \) is an outer boundary. Equation (1) can be assembled and represented in matrix form after row and column ordering [1] \( \sum_{i=1}^{N} H_i x_i = c_i \) where \( x_i, \; i=1,\ldots,N \), are column vectors representing the unknowns in the domains.

2. DOMAIN COUPLING

Contacting points are typically coupled by either utilising a penalty method or by means of Lagrangian multipliers [2]. However, it is possible and beneficial to capture these contact conditions with

\[
\begin{align*}
p_{n}^i(\tilde{x}) &= -p_{n}^i(\tilde{x}) = -k_{n}^y(\tilde{x})\Delta L_{n}^y(\tilde{x}) + k_{n}^y(\tilde{x})\left[u_{n}^i(\tilde{x}) - u_{n}^j(\tilde{x})\right] \quad (2a) \\
p_{t}^i(\tilde{x}) &= -p_{t}^i(\tilde{x}) = -k_{t}^y(\tilde{x})\Delta L_{t}^y(\tilde{x}) + k_{t}^y(\tilde{x})\left[u_{t}^i(\tilde{x}) - u_{t}^j(\tilde{x})\right] \quad (2b)
\end{align*}
\]

where \( m = 1, 2 \), \( \Delta L^y(\tilde{x}) \) is the initial gap, subscripts \( n \) and \( t \) refers to normal and tangent directions, respectively. The use of conditions (2) introduces additional unknowns into the problem, i.e. the interfacial spring stiffness coefficients \( k_{n}^y(\tilde{x}) \). In practice, however, these are easily set in the context of the problem under consideration. Relatively high values of \( k_{n}^y(\tilde{x}) \) will achieve high accuracy in the matching of displacement contact conditions. This may not be required however, since in many cases, it is possible to accept a certain degree of mismatch at an interface without significant loss of accuracy.

3. COARSE PRECONDITIONER

Direct communication between domains can be beneficial to iterative solvers [1]. Consider again equation (1), applied to the \( i \)th domain subjected to thermal loading, i.e.

\[
c_{\alpha} u_{\alpha}^i + \sum_{c=i-q}^{i+q} \int_{\Delta_{\alpha}} N_q p_{\alpha}^* d\Gamma - \sum_{c=i-q}^{i+q} \int_{\Delta_{\alpha}} N_q u_{\alpha}^* d\Gamma - \sum_{c=i-q}^{i+q} \int_{\Delta_{\alpha}} N_q T_{\alpha}^* d\Gamma = -\gamma' \int_{\Gamma^i} \phi \phi^i n_1 d\Gamma - \gamma' \int_{\Gamma^i} T_{\alpha} n_1 d\Gamma \quad (3)
\]
where the thermal field is Laplacian. Let \( \Gamma^i = \Gamma^i_0 \cup \bigcup_{j \neq i} \Gamma^i_j \) be the boundary for \( \Omega^i \), where \( \Gamma^i_j \subset \Gamma^i \) is an interfacing face between \( \Omega^i \) and \( \Omega^j \), and \( \Gamma^i_0 \subset \Gamma^i \) denotes an external surface not interfacing with another domain. Consider further the subdivision of \( \Gamma^i_0 \) into regions defined by their boundary conditions, i.e., \( \Gamma^i_0 = \Gamma^i_{0\ell} \cup \Gamma^i_{0\kappa} \cup \Gamma^i_{0\alpha} \), where \( p_k^i = \overline{p}_k \) on \( \Gamma^i_{0\ell} \), \( p_k^i = \overline{k}_k(\overline{u}_k^i - \overline{u}_k^\alpha) \) on \( \Gamma^i_{0\kappa} \) and \( \overline{u}_k^i = \overline{u}_k^\alpha \) on \( \Gamma^i_{0\alpha} \). It is convenient at this stage to adjust the notation so that \( \Gamma^i_j = \bigcup_{\parallel j \neq i} \Gamma^i_0 \), where subdivisions of \( \Gamma^i_0 \) are identified by \( \Gamma^i_j \) for \( j \neq i \) and \( j \in \{N_1 + \ell : \ell = 1,2,3\} \). Consider \( m^i_j \) distinct points placed at selected nodes on the surface \( \Gamma^i_j \). In addition, consider the partition of \( \Gamma_j^i \) into \( m^i_j \) surfaces \( \Gamma^i_\ell \) such that \( \Gamma^i_j = \bigcup_{\parallel \ell = 1,2,3} \Gamma^i_\ell \), and where \( \Gamma^i_\ell \) is a collection of elements on \( \Gamma^i_j \). Consider the following approximations

\[
\int_{\Gamma^i_\ell} p_k^i u_k^i \, d\Gamma = \sum_{c=1}^{\infty} \sum_{k=1}^{N_\ell} u_k^i \overline{N}_q^i p_k \, d\Gamma \quad \text{and} \quad \int_{\Gamma^i_\ell} u_k^i p_k^i \, d\Gamma = \sum_{c=1}^{\infty} \sum_{k=1}^{N_\ell} p_k^i \overline{N}_q^i u_k^\ell \, d\Gamma \tag{4}
\]

\[
\int_{\Gamma^i_\ell} T^i \phi_{i\ell} n_k \, d\Gamma = \sum_{c=1}^{\infty} \sum_{k=1}^{N_\ell} \overline{T}^i \phi_{i\ell} \overline{n}_k \, d\Gamma \quad \text{and} \quad \int_{\Gamma^i_\ell} \phi_{i\ell} T^i n_k \, d\Gamma = \sum_{c=1}^{\infty} \sum_{k=1}^{N_\ell} \phi_{i\ell} \overline{T}^i \overline{n}_k \, d\Gamma \tag{5}
\]

where the superscript a denotes approximation, and where \( N_\ell \) are shape functions, \( E_\ell^i \) is the number of elements on \( \Gamma^i_\ell \) and \( N_\ell \) is the number of nodes on an element. The idea here is to obtain a relatively crude approximation, where typically only a small number of points are placed local to each surface \( \Gamma^i_\ell \), giving rise to a relatively small system of the form \( \sum_{i=1}^{N_\ell} H^i x^i = c^\ell \), in assembled form

\[
H^e x^e = c^e \quad \text{with} \quad n^e \text{ unknowns, where superscript 'a' denotes approximation. In principle this system can be solved as } H^e \text{ is constrained to be invertible. In order to establish a relationship between } c^e \text{ and } c \text{ equations (4) are equated with the approximations used in the construction of } \sum_{i=1}^{N_\ell} H^i x^i = c^\ell, \text{ which gives an assembled system of the form } c^e = H^e x^e = \tilde{H} x^\ell, \text{ where } \tilde{H} \text{ is an } n^e \times n \text{ matrix. Note that, } \tilde{H} \text{ consists of those rows of } H \text{ associated with the coincident node. Consider then, } c^e = \tilde{H} H^{-1} c = R_c \text{, where } R \text{ is an } n^e \times n \text{ rectangular matrix. The approximation } D x = E^T x^e + c, \text{ where } E \text{ is an } n^e \times n \text{ matrix and } D \text{ is an } n \times n \text{ non-singular diagonal matrix connects } x^e \text{ and } x, \text{ and is obtained by application of } c_{\ell\ell} u^\ell_{\ell} = -\int_{\Gamma^i_\ell} \overline{p}_k^\ell \overline{u}^\ell_k \, d\Gamma + \int_{\Gamma^i_\ell} u_k^\ell p_k^\ell \, d\Gamma + \gamma^\ell \int_{\Gamma^i_\ell} T^i_{\ell\ell} \phi_{i\ell} n_k \, d\Gamma - \gamma^\ell \int_{\Gamma^i_\ell} \phi_{i\ell} T^i_{\ell\ell} n_k \, d\Gamma \text{ at all points on } \Gamma^i_\ell.
the fine mesh. The above approximations enable an outer loop correction to the solution schemes of the form, \( x^{(k+i)} = x^{(k)} + D^{-1}\left(E^T H^{-1} R + I\right)c - Hx^{(k)} \). In section 4 this relatively crude approach is shown to be sufficient to significantly enhance the convergence of the iterative schemes.

4. NUMERICAL EXPERIMENTATION

In order to demonstrate the benefit of the coarse preconditioner a two domain model was tested. A three-dimensional thermoelastic BE model used for predicting die deformation in the pressure die casting process is employed [3]; linear triangular elements are utilised. Figure 1(a) shows an isometric view of the surface mesh of the model which generates 4899 equations. The die interfaces are connected via springs of stiffness coefficient 20 kN/mm. The tests were carried out on a SUN BLADE 1000. The initial trial vector \( x^{(0)} \) is set to 0, which is sufficiently remote from the exact solution. The system termination criteria of

- \( \left\| d_1 - Hx_i \right\|_2 \times \left\| d_i \right\|_2^{-1} < 1 \times 10^{-5} \) was utilised for individual domains and
- \( \left\| c - Hx \right\|_2 \times \left\| d_i \right\|_2^{-1} < 1 \times 10^{-4} \) for the entire system, where \( d_i \) refers to the constant identified with domain \( i \) and includes coupling information from other domains.

The following schemes [1] are considered:

1. LU decomposition with forward and back substitution.
2. Gauss-Seidel iteration.
3. Generalised Successive Overrelaxation (GSOR),
4. GMRES Jacobi preconditioning.
5. GMRES with parameter matrix preconditioning.

Table 1 shows the results obtained with and without the implementation of the coarse preconditioner. Focusing on the tests without the preconditioner (WCP) it is apparent that the total number of domain (outer) iterations approaches that of LU decomposition with increase in the number of inner iterations. This is anticipated because the residual error associated with each domain should reduce to that of the direct method with increase in number of inner iterations. Although the Gauss-Seidel and GSOR converge in less iterations in some cases, when the iterations per domain are set to one or two, this can be attributed to the termination procedure.

Consider now the tests performed to investigate the performance of the coarse preconditioner described in section 3. All nodes on the interface surfaces and a single node on all other surfaces were employed in the construction of the coarse preconditioner. The model generated 1326 coarse system equations with the stiffness coefficient \( k \) set at 20 kN/mm. The results from the test are given in Table 1 (CP), where upon inspection it can be seen that the application of the coarse preconditioner resulted, in general, in a large reduction in the number of block and total iterations required for convergence. The best performances of the iterative schemes were obtained when the number of iterations permitted per domain was set to 5 and 10. The fastest solution times were obtained by the PC GMRES, which was 8 time faster than LU decomposition. A further set of tests, designed to investigate the effect of stiffness on the application of the coarse
preconditioner to the solution of multi-domain systems were performed. The effect of three different stiffness, $k$, were investigated, namely 1, 10, and 20 kN/mm$^3$. It was concluded that the higher the stiffness the more block and total iterations are needed to obtain the solution.

5. FLASH PREDICTION IN PRESSURE DIE CASTING

In this section a methodology for modelling the interaction between die interfaces is presented, which is particularly useful for predicting flash. The die blocks are linked using springs as a mechanism for modelling the interaction between them. Iteration is necessary to cater for the determination of the unknown contact conditions at the block interfaces. This new methodology is applied to a real industrial die that has been used in extensive experimental tests concerned with the modelling and prediction of flash [3]. The meshes of the die blocks and casting for this problem are shown in Figure 1(a). The normal boundary condition of interest is

$$p_n(x) = -p_n^e(x) = -k_{12}^n \Delta L_{12}^n(x) + k_{12}^n (u_1^n(x) - u_2^n(x)),$$

where the interfacial stiffness coefficient $k_{12}^n$ is assumed to be independent of position $x$ at the interface and $\Delta L_{12}^n(x)$ is the normal die separation distance. An initial value for $\Delta L_{12}^n$ is taken to be 40 $\mu$m. A value of $k_{12}^n = 0.4$ kN/mm$^3$ was found to be sufficient to give

$$\left| -\Delta L_{12}^n + (u_1^n(x) - u_2^n(x)) \right| < 1\mu m$$

at the contacting points. A simple bisection method was utilised to adjust $\Delta L_{12}^n$ so that the total reaction forces were within 5% of the force imposed by the clamping system, which was 2 Tonnes. Springs were added or removed according to the sign of $-\Delta L_{12}^n + (u_1^n(x) - u_2^n(x))$. The algorithm converged in less than 6 iterative adjustments to $\Delta L_{12}^n$. Figure 1(b) [3] shows a photograph of a component exhibiting flash, produced under the operating conditions given in reference [3]. Figure 1(c) shows the regions where the die interfaces are predicted to be open by greater than 30 $\mu$m. Upon comparison of Figures 1(b) and 1(c) it can be seen that the contact methodology has predicted gaps in precisely those areas where flash was observed in the experimental tests on the die. Coarse preconditioning of the PC GMRES method resulted in factors of reduction in CPU time and block iterations of 2.54 and 3.33 respectively.

6. CONCLUSIONS

From this paper it can be concluded that the coarse preconditioner enhances the convergence of multi-domain systems. The minimum number of iterations was obtained when all nodes on the interfaces with high stiffness coefficients are included in the coarse model.
REFERENCES


Figure 1. Mesh, experimentally observed and predicted flash on casting.

<table>
<thead>
<tr>
<th>Max</th>
<th>Number of Block Gauss-Seidel Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>LU Decomp</td>
</tr>
<tr>
<td></td>
<td>WCP</td>
</tr>
<tr>
<td>1</td>
<td>56 (112)</td>
</tr>
<tr>
<td>2</td>
<td>58 (232)</td>
</tr>
<tr>
<td>5</td>
<td>56 (559)</td>
</tr>
<tr>
<td>10</td>
<td>56 (884)</td>
</tr>
</tbody>
</table>

WCP & CP-Without and With Application of Coarse Precondition Respectively. Max-Maximum number of iterations per domain.

Table 1. Total number of block and single domain iterations for test.
Evaluation of logarithmic integrals in two-dimensional boundary element computation
D. Crann*, A.J. Davies*, D.B. Christianson*

*Department of Mathematics, *Department of Computer Science,
*University of Hertfordshire, Hatfield, AL109AB
{d.crann, a.j.davies, b.christianson}@herts.ac.uk

Keywords: Boundary elements, logarithmic integrals, automatic differentiation

Abstract
In the development of the system matrices for two-dimensional boundary element computations there is a necessity to evaluate logarithmic singular integrals. For linear elements these singular integrals may be developed analytically. However, for quadratic elements this is not the case and a numerical quadrature is required. We shall consider four possible approaches to the evaluation of the singular integrals which will be compared in terms of accuracy, efficiency and ease of implementation.

Introduction
The system of equations which arises in the collocation approach to the boundary element method requires the evaluation of integrals over the so-called target elements whose integrals are functions of $R$, the distance from the so-called base node (the collocation node). When the base node is not in the target element the integrals are non-singular and may be evaluated using standard Gauss-Legendre quadrature. It is possible in regions with high aspect ratio for $R$ to be small, but non-zero. In these cases we have nearly singular integrals and we need to take particular care and there are some schemes especially developed for such cases [1,2]. If the base node is in the target element then the integrals are singular and in this case there is a variety of ways in which the integrals may be evaluated. Crann et al. [3] describe these methods and introduce the automatic differentiation approach. For constant or linear elements the logarithmic integrals may be evaluated analytically [4] as they can for elements with linear geometry and quadratic potential variations [5,6,7]. For isoparametric quadratic elements we require a suitable numerical approach.

Logarithmic integrals
We shall develop the results with respect to the quadratic elements whose nodes have position vectors
We choose the local coordinates \( \{ \xi : -1 \leq \xi \leq 1 \} \) to parameterise the quadratic interpolation of the approximate boundary curve through \( \mathbf{r}_i \) via the Lagrange interpolation polynomials
\[
L_1(\xi) = \frac{1}{2} \xi (\xi - 1), \quad L_2(\xi) = 1 - \xi^2, \quad L_3(\xi) = \frac{1}{2} \xi (\xi + 1)
\]
The element geometry is then defined by
\[
\mathbf{r}(\xi) = \sum_{i=1}^{3} L_i(\xi) \mathbf{r}_i
\] (1)

If the base node is \( \mathbf{r}_j \), then we write \( \mathbf{R}_j(\xi) = \mathbf{r}(\xi) - \mathbf{r}_j \), the position vector of a point in the target element relative to the base node. We then require the evaluation of nine singular integrals
\[
I_j = \int_{-1}^{1} L_i(\xi) J(\xi) \ln R_j(\xi) d\xi \quad i, j = 1, 2, 3
\] (2)

where \( J(\xi) \) is the Jacobian of the transform given by
\[
J(\xi) = \left[ \mathbf{r}'(\xi) \cdot \mathbf{r}'(\xi) \right]^{\frac{1}{2}}
\]

Now suppose that the singularity is at \( \xi = \xi_0 \) i.e. \( \mathbf{r}(\xi_0) = \mathbf{r}_j \).

Then \( R_j(\xi) = \left| \mathbf{r}(\xi) - \mathbf{r}_j \right| \)
\[
= \left| \mathbf{r}'(\xi_0) \Delta \xi + \frac{1}{2} \mathbf{r}'(\xi_0) \Delta \xi^2 \right| = \left| \Delta \xi \right| d_1 \Delta \xi + d_2 \Delta \xi^2
\]

with \( \Delta \xi = \xi - \xi_0 \), where \( \xi_0 = -1, 0 \) or \(+1\).

It follows that
\[
I_j = \int_{-1}^{1} L_i(\xi) J(\xi) \ln \left| \Delta \xi \right| d\xi + \frac{1}{2} \int_{-1}^{1} L_i(\xi) J(\xi) \ln \left| R_j(\xi) \right| d\xi
\] (3)

Log-Gauss quadrature
The first integral in equation (3) is transformed, via a change of variable \( \xi \rightarrow \eta \), to an integral of the form
\[
\frac{1}{2} \int_{-1}^{1} f(\eta) \ln |\eta| d\eta
\] (4)

which may be evaluated using a special logarithmic Gauss quadrature [8].

The second integral is performed using a standard Gauss quadrature.

---

ISBN 0904 188965
Telles transformation
A transformation \( \xi \rightarrow \eta \), where \( \xi \) is cubic in \( \eta \), is applied to the integral in such a manner that the Jacobian, \( \xi'(\eta) \), is zero at \( \xi_0 \), \( \xi(\eta_0) \) say. \( I_j \) takes the form

\[
\int_{\eta_1}^{\eta} F(\eta)\xi'(\eta)\ln R_j(\xi(\eta))d\eta
\]

where the integrand is now zero at \( \eta_0 \) and may be evaluated using a standard Gauss quadrature.

Automatic differentiation
We return to the integral in equation (3) and expand each term in the integrand as a Taylor polynomial of degree \( n \) \([3]\) which leads to an expression of the form

\[
I_j = \int_{\eta_1}^{\eta} \sum_{k=0}^{n} a_k \Delta \xi^k \ln |\Delta \xi| d\xi + \int_{\eta_1}^{\eta} \sum_{k=0}^{n} b_k \Delta \xi^k d\xi
\]

\[
= \sum_{k=0}^{n} (a_k \alpha_k + b_k \beta_k)
\]

where \( \alpha_k = \int_{\eta_1}^{\eta} \Delta \xi^k \ln |\Delta \xi| d\xi \), \( \beta_k = \int_{\eta_1}^{\eta} \Delta \xi^k d\xi \).

Each of these integrals may be evaluated analytically and the values of \( \alpha_k \) and \( \beta_k \) for the three cases are given by Crann et al. \([3]\). We note here that the Taylor polynomials come from sequences of truncated Taylor series and we must ensure that the series converges as the number of terms is increased. Crann et al. \([9]\) show that the condition on the positioning of the nodes must be such that an estimate of the radius of convergence is given by

\[
\rho = \min \left[ \frac{|r_1 - r_2|}{2|r_2 - (r_1 - r_2)|} \right]
\]

A value of \( \rho > 3 \) is likely to be satisfactory and all likely boundary discretisations would have \( \rho \) significantly larger than 3.
Beale and Attwood’s formula

An interesting quadrature procedure has been suggested by Beale and Attwood [10] in which the logarithmic integrals are evaluated as follows:

\[ \int_{-1}^{1} f(t) \ln|t| \, dt = \sum_{j=-N}^{j=N} f(jh) \ln|jh| w_j h + h \ln\left(\frac{1}{h}\right) f(0) \]

where

\[ w_j = 1, \quad j = -N + 2, \ldots, N - 2; \quad w_{-N+1} = w_{N-1} = 13/12; \quad w_{-N} = w_N = 5/12. \]

Numerical examples

We consider the curved element with nodes

\[ r_1 = (1,0) \quad r_2 = \left(0.5 + \alpha \sqrt{2}, 0.5 + \alpha \sqrt{2}\right) \quad r_3 = (0.1) \]

In the tables which follow we compare the results with an accurate result obtained using the symbolic computation package Maple®. The tables show the absolute value of the integral for each method, Log-Gauss 10 point, Telles 20 point, A-D 20 degree and Beale and Attwood’s method with \( N = 1000 \).

<table>
<thead>
<tr>
<th>( \alpha = 0.0 )</th>
<th>( I_{11} = I_{33} )</th>
<th>( I_{12} = I_{32} )</th>
<th>( I_{13} = I_{31} )</th>
<th>( I_{21} = I_{23} )</th>
<th>( I_{22} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maple</td>
<td>0.5861349</td>
<td>0.4589215</td>
<td>0.1209719</td>
<td>0.1602556</td>
<td>1.583831</td>
</tr>
<tr>
<td>L-G 10pt</td>
<td>0.5861339</td>
<td>0.4589211</td>
<td>0.1209707</td>
<td>0.1602562</td>
<td>1.583830</td>
</tr>
<tr>
<td>Telles 20pt</td>
<td>0.5861350</td>
<td>0.4589215</td>
<td>0.1209719</td>
<td>0.1602556</td>
<td>1.584883</td>
</tr>
<tr>
<td>A-D 20deg</td>
<td>0.5861349</td>
<td>0.4589214</td>
<td>0.1209719</td>
<td>0.1602556</td>
<td>1.583831</td>
</tr>
<tr>
<td>Beale 1000pt</td>
<td>0.5861375</td>
<td>0.4589165</td>
<td>0.1209709</td>
<td>0.1602555</td>
<td>1.583832</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>( \alpha = 0.001 )</th>
<th>( I_{11} = I_{33} )</th>
<th>( I_{12} = I_{32} )</th>
<th>( I_{13} = I_{31} )</th>
<th>( I_{21} = I_{23} )</th>
<th>( I_{22} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maple</td>
<td>0.5861380</td>
<td>0.4589190</td>
<td>0.1209726</td>
<td>0.1602563</td>
<td>1.583833</td>
</tr>
<tr>
<td>L-G 10pt</td>
<td>0.5861370</td>
<td>0.4589188</td>
<td>0.1209714</td>
<td>0.1602568</td>
<td>1.583832</td>
</tr>
<tr>
<td>Telles 20pt</td>
<td>0.5861381</td>
<td>0.4589190</td>
<td>0.1209726</td>
<td>0.1602563</td>
<td>1.584885</td>
</tr>
<tr>
<td>A-D 20deg</td>
<td>0.5861380</td>
<td>0.4589190</td>
<td>0.1209726</td>
<td>0.1602563</td>
<td>1.583833</td>
</tr>
<tr>
<td>Beale 1000pt</td>
<td>0.5861439</td>
<td>0.4589244</td>
<td>0.1209719</td>
<td>0.1602561</td>
<td>1.583834</td>
</tr>
</tbody>
</table>

ISBN 0904 188965
We give results for $\alpha$ in the range $0 \leq \alpha \leq 0.2$, since in any reasonable discretisation $\alpha$ would not be outside this range, in fact it would be unlikely to have $\alpha$ as large as 0.1. We see, as may be expected, that in general accuracy decreases as $\alpha$ increases and this is much more pronounced for the A-D and Beale and Attwood’s methods.

In table 6 we show the operation count for each method and we see that the Log-Gauss integral requires significantly fewer operations than the others.

**Table 3 $\alpha = 0.01$**

<table>
<thead>
<tr>
<th></th>
<th>$I_{11} = I_{33}$</th>
<th>$I_{12} = I_{32}$</th>
<th>$I_{13} = I_{31}$</th>
<th>$I_{21} = I_{23}$</th>
<th>$I_{22}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maple</td>
<td>0.5864408</td>
<td>0.4586762</td>
<td>0.1210427</td>
<td>0.1603226</td>
<td>1.584021</td>
</tr>
<tr>
<td>L-G 10pt</td>
<td>0.5864398</td>
<td>0.4586761</td>
<td>0.1210415</td>
<td>0.1603232</td>
<td>1.584020</td>
</tr>
<tr>
<td>Telles 20pt</td>
<td>0.5864410</td>
<td>0.4586762</td>
<td>0.1210427</td>
<td>0.1603226</td>
<td>1.585073</td>
</tr>
<tr>
<td>A-D 20deg</td>
<td>0.5864408</td>
<td>0.4586763</td>
<td>0.1210427</td>
<td>0.1603226</td>
<td>1.584021</td>
</tr>
<tr>
<td>Beale 1000pt</td>
<td>0.5867881</td>
<td>0.4596170</td>
<td>0.1210743</td>
<td>0.1603227</td>
<td>1.584030</td>
</tr>
</tbody>
</table>

**Table 4 $\alpha = 0.1$**

<table>
<thead>
<tr>
<th></th>
<th>$I_{11} = I_{33}$</th>
<th>$I_{12} = I_{32}$</th>
<th>$I_{13} = I_{31}$</th>
<th>$I_{21} = I_{23}$</th>
<th>$I_{22}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maple</td>
<td>0.6140498</td>
<td>0.4351688</td>
<td>0.1277605</td>
<td>0.1659606</td>
<td>1.601812</td>
</tr>
<tr>
<td>L-G 10pt</td>
<td>0.6140489</td>
<td>0.4351685</td>
<td>0.1277617</td>
<td>0.1659600</td>
<td>1.601811</td>
</tr>
<tr>
<td>Telles 20pt</td>
<td>0.6140499</td>
<td>0.4351687</td>
<td>0.1277617</td>
<td>0.1659600</td>
<td>1.602864</td>
</tr>
<tr>
<td>A-D 20deg</td>
<td>0.6138296</td>
<td>0.4354450</td>
<td>0.1277488</td>
<td>0.1659600</td>
<td>1.601812</td>
</tr>
<tr>
<td>Beale 1000pt</td>
<td>0.6450671</td>
<td>0.5220976</td>
<td>0.1307252</td>
<td>0.1659603</td>
<td>1.602733</td>
</tr>
</tbody>
</table>

**Table 5 $\alpha = 0.2$**

<table>
<thead>
<tr>
<th></th>
<th>$I_{11} = I_{33}$</th>
<th>$I_{12} = I_{32}$</th>
<th>$I_{13} = I_{31}$</th>
<th>$I_{21} = I_{23}$</th>
<th>$I_{22}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maple</td>
<td>0.6761561</td>
<td>0.3706488</td>
<td>0.1456680</td>
<td>0.1744901</td>
<td>1.646123</td>
</tr>
<tr>
<td>L-G 10pt</td>
<td>0.6761551</td>
<td>0.3706487</td>
<td>0.1456669</td>
<td>0.1744908</td>
<td>1.646122</td>
</tr>
<tr>
<td>Telles 20pt</td>
<td>0.6761562</td>
<td>0.3706488</td>
<td>0.1456680</td>
<td>0.1744901</td>
<td>1.647175</td>
</tr>
<tr>
<td>A-D 20deg</td>
<td>0.0898890</td>
<td>0.5363525</td>
<td>0.4744941</td>
<td>0.1735002</td>
<td>1.650219</td>
</tr>
<tr>
<td>Beale 1000pt</td>
<td>0.7757384</td>
<td>0.6610460</td>
<td>0.1553079</td>
<td>0.1744903</td>
<td>1.649277</td>
</tr>
</tbody>
</table>

**Table 6 Operation count for each method**

<table>
<thead>
<tr>
<th></th>
<th>L-G 10pt</th>
<th>Telles 20pt</th>
<th>A-D 20deg</th>
<th>Beale 1000pt</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ -</td>
<td>1,431</td>
<td>4,169</td>
<td>38,436</td>
<td>153,089</td>
</tr>
<tr>
<td>* /</td>
<td>1,962</td>
<td>7,634</td>
<td>15,051</td>
<td>210,139</td>
</tr>
<tr>
<td>sqrt, log</td>
<td>162</td>
<td>362</td>
<td>138</td>
<td>15,012</td>
</tr>
<tr>
<td>Total</td>
<td>3,555</td>
<td>12,165</td>
<td>53,625</td>
<td>378,240</td>
</tr>
</tbody>
</table>
In terms of ease of implementation we note first that the A-D approach would be adopted only in an environment which supports operator overloading and Taylor series data-types so a general user would be unlikely to adopt it. However, its attraction to users is that the errors are due only to truncation errors in the Taylor series and not to a numerical quadrature rule. For smaller values of $\alpha$ A-D gives the best accuracy. For the other three methods implementation costs are very similar. We also note here that Beale and Attwood’s method is interesting because it does not require a data set of quadrature points which depend on the order of quadrature.

Comparing the four methods, in general we see that the log-Gauss method provides the best approach in terms of accuracy, efficiency and ease of implementation.

References
Boundary Interpolation Instead of Boundary Integrals: A Meshless Approach

Csaba Gáspár

Széchenyi István University, Department of Mathematics, P.O.Box 701, H-9007 Gyor, Hungary
email: gasparcs@mail.sze.hu

Keywords: meshless methods, boundary interpolation, radial basis functions

Abstract. A meshless boundary method based on a boundary interpolation technique is presented, which completely avoids the use of boundary integrals as well as large and dense matrices. The interpolation is defined by solving a higher-order auxiliary partial differential equation supplied with the interpolation conditions as special boundary conditions. To solve this partial differential equation, robust multi-level quadtree-based algorithms are used, requiring much less computational cost than the traditional BEM solvers. The approach is presented for the 2D Laplace equation and the 2D Stokes problem.

Introduction

The classical boundary integral equation method converts a domain problem to a boundary one. This makes it possible to avoid the domain discretisation and to reduce the dimension of the original problem. However, as it is well known, the resulting boundary integral equations generally lead to full, nonsymmetric and often ill-conditioned system of algebraic equations. Moreover, if the original problem is inhomogeneous and/or nonlinear, domain integrals also appear in the boundary integral equations. To evaluate these domain integrals without introducing domain discretisation, the usual way is to apply a scattered data interpolation method and dual reciprocity. Using the popular method of radial basis functions (RBFs, see [1]), one has to face some computational disadvantages similar to that of the traditional boundary integral equation method, since again large, dense and ill-conditioned linear systems have to be solved.

We present here an approach which circumvents the above problem. The method is based on the polyharmonic-type interpolation proposed in [2,3]. In this approach, the interpolation function is assumed to satisfy a higher-order partial differential equation (typically an iterated Helmholtz-like equation) including the interpolation conditions as special pointwise boundary conditions. It has turned out that this results in an RBF-like interpolation based on the fundamental solution of the applied partial differential operator. However, the RBFs
are no used in an explicit way. Instead, it is the higher order auxiliary partial differential equation that has to be solved. Since the domain of this partial differential equation is practically arbitrary, fast and robust methods can be applied based on the quadtree subdivision and multi-level techniques [4]. Using the principle of particular solutions (see e.g. [5]), the domain integrals (Newtonian potentials) can be evaluated in a fast and meshless way by converting it to an additional homogeneous boundary problem without source terms. The remaining boundary problem is solved by using a boundary interpolation with a carefully chosen auxiliary partial differential operator. This operator is defined in such a way that the interpolation is sufficiently exact along the boundary and approximates the desired partial differential operator far from the boundary. This results in a method which is completely meshless but still approximates the exact solution satisfactorily. The method works well in case of Dirichlet problems for the Poisson equation [3]. In this paper, we extend the approach for 2D Stokes problems as well.

Polyharmonic-type interpolation

The applied boundary interpolation method is based on the scattered polyharmonic-type (or multi-elliptic) interpolation [2]. Let \( \Omega_0 \) be a bounded, smooth domain of the plane (note that similar results are valid for the three-dimensional space as well). If the points \( x_1, x_2, \ldots, x_N \in \Omega_0 \) and the associated values \( u_1, u_2, \ldots, u_N \in \mathbb{R} \) are given, one can define an interpolation function \( u : \Omega_0 \rightarrow \mathbb{R} \) as a solution of a partial differential equation with a properly chosen partial differential operator \( L \):

\[
Lu = 0 \quad \text{in} \quad \Omega_0 \setminus \{x_1, x_2, \ldots, x_N\}
\]

(the equality is understood in the sense of distributions). Along the boundary \( \Gamma_0 := \partial \Omega_0 \), a regular boundary condition e.g. a homogeneous Dirichlet boundary condition is imposed, while at the interpolation points \( x_1, x_2, \ldots, x_N \), the interpolation conditions are prescribed:

\[
u(x_k) := u_k \quad (k = 1, 2, \ldots, N).
\]

The interpolation conditions can be regarded as special boundary conditions taken at discrete points. Note that, such pointwise boundary conditions make the usual second-order partial differential equations ill-posed. For higher order equations, however, this does not remain the case. The well-posedness depends on the fact whether or not the actual function space (in which the problem is defined) contains the continuous functions i.e. whether or not the Dirac func-
tional concentrated on the interpolation points are bounded with respect to the actual norm.

The simplest and most natural choice for $L$ is the biharmonic operator: $Lu := \Delta^2 u$. In this case, the problem give by eqs (1) - (2) can be reformulated either in a direct or in a variational form as follows.

The direct problem: For a given function $u_0 \in W$, where $W := \{ v \in H^2_0(\Omega_0) : v(x_1) = v(x_2) = \ldots = v(x_N) = 0 \}$ (note that $W$ is a closed subspace of the Sobolev space $H^2_0(\Omega_0)$ due to the embedding theorems), find a function $w \in W$ such that (in the sense of distributions):

$$\Delta^2(u_0 + w) = 0 \quad \text{in} \quad \Omega_0 \setminus \{x_1, x_2, \ldots, x_N\}$$  \hspace{1cm} (3)

The variational problem: For a given function $u_0 \in W$, find a function $w \in W$ such that for every $v \in W$, the following equality is satisfied:

$$\langle \Delta(u_0 + w), \Delta v \rangle_{L^2(\Omega_0)} = 0$$  \hspace{1cm} (4)

The above two problems are equivalent and have unique solutions [2]. In both cases, the interpolation function is given by $u := u_0 + w$, which can be uniquely represented in the following form:

$$u(x) = w(x) + \sum_{j=1}^{N} \beta_j \Phi(x - x_j),$$  \hspace{1cm} (5)

where $w$ is an everywhere biharmonic function (including also the interpolation points) and $\Phi(x) = (\| x \|^2 \log \| x \|)/8\pi$ is the biharmonic fundamental solution.

Eq (5) is an expression which is similar to the interpolation form of the method of radial basis functions (completed with a regular function $w$). However, in order to compute the coefficients $\beta_1, \beta_2, \ldots, \beta_N$, no large linear algebraic system has to be solved. Instead, the interpolation function $u$ should be determined by solving either the direct problem (3) or the variational problem (4). Once one of them has been solved, the coefficients in eq (5) can be determined by evaluating the following expression which is a direct consequence of eq (5):

$$\Delta \Delta u = \sum_{j=1}^{N} \beta_j \delta_{x_j}.$$  \hspace{1cm} (6)

Here $\delta_{x_j}$ denotes the Dirac functional concentrated on the point $x_j$. Roughly speaking, $\beta_j$ is the intensity of the weak singularity of $\Delta \Delta u$ at the point $x_j$. 

Advances in Boundary Element Techniques 329
Eds R Gallego & M H Aliabadi  Copyright 2003
To solve the direct or the variational problem (eqs (3), (4)), quadtree/octree-based multilevel methods can be recommended because of their robustness [4]. Note that, in real interpolation problems, the domain $\Omega_0$ can be defined in a practically arbitrary way, therefore the computational cost of the implementation of the method can be kept at a moderate level. The number of the necessary algebraic operations is typically $O(N \log N)$ only, which is much less than that of the traditional RBF-methods.

The approach can be generalized in various ways. It is possible to use polyharmonic operators: $Lu := \Delta^m u$ ($m > 2$, an integer); in this case, the interpolation function is sought in an appropriate closed subspace of the Sobolev space $H^m_0(\Omega_0)$. It is also possible to use the elliptic operator $\Delta - c^2 I$ instead of the Laplacian, where $c > 0$ is a scaling constant and $I$ denotes the identity operator. For instance, the operator $L := (\Delta - c^2 I)^2$ results in a method which is very similar to the biharmonic interpolation. The essential difference between them is that while the biharmonic fundamental solution is globally supported, the fundamental solution of $(\Delta - c^2 I)^2$ is rapidly decreasing far from the origin: $\Phi(x) = (c \|x\| \cdot K_1(c \|x\|) / 4c^2 \pi$. Here $K_1$ denotes the usual modified Bessel function. Consequently, $\Phi$ behaves as if it were compactly supported. This simplifies the representation (5) by eliminating the term $w$ (provided that $\Omega_0$ is large enough). The size of the essential support of $\Phi$ can be controlled by the parameter $c$ and should be (several times) bigger than the separation distance of the interpolation points.

A general formulation of such kind of operators $L$ is as follows:

$$L := \prod_{k=1}^{m}(\Delta - c_k^2 I),$$

allowing some $c_k$ to be zero. If each $c_k$ is greater that 0, the fundamental solution of $L$ is rapidly decreasing. Otherwise, it approximates the fundamental solution of a lower order partial differential operator in some sense. This makes it possible to construct boundary interpolation methods as discussed below.

**Boundary interpolation techniques**

Suppose that all the interpolation points are located along the boundary $\Gamma$ of a smooth subdomain $\Omega$ of $\Omega_0$. Then the behavior of the interpolation function $u$ along $\Gamma$ is essentially determined by the interpolation conditions. However, the behavior of $u$ inside $\Omega$ (i.e. far from the boundary) is mainly determined by the
asymptotic behavior of the fundamental solution of the applied operator $L$ (far from the origin). Therefore $u$ may (approximately) satisfy another partial differential equation inside the domain, apart from a narrow vicinity of the boundary.

Based on this observation, let us choose the interpolation operator in the following form (mixed Laplace-Helmholtz-interpolation):

$$L := \Delta(\Delta - c^2 I).$$

Then the associated fundamental solution is

$$\Phi(x) = -(K_0(c \| x \|) + \log(c \| x \|))/2c^2 \pi,$$

where $K_0$ the usual modified Bessel function. The scaling parameter $c$ should be chosen in such a way, that the size of the essential support of $K_0(c \| x \|$) is in the same order of magnitude that the separation distance of the neighboring boundary points. Then the interpolation function becomes nearly harmonic inside $\Omega$ (except a narrow neighborhood of $\Gamma$) and approximates a predefined function $u_0$ along $\Gamma$, i.e. it approximates the solution of the Dirichlet problem:

$$\Delta u = 0 \text{ in } \Omega, \quad u = u_0 \text{ along } \Gamma$$

Application to the 2D Stokes problem: The 2D steady Stokes problem is governed by the following equations (in dimensionless form):

$$\text{div } \mathbf{u} = 0, \quad -\Delta \mathbf{u} + \text{grad}p = 0.$$}

Using the well-known stream function approach, the problem can be converted to the biharmonic equation

$$\Delta^2 \psi = 0,$$

where $\psi$ denotes the stream function: $\mathbf{u} = (\partial \psi / \partial y, -\partial \psi / \partial x)$. Eq (11) is often supplied with the Neumann type boundary condition:

$$\text{grad} \psi = \mathbf{v} \text{ along } \Gamma,$$

i.e. the velocity vector is prescribed along the boundary.

The solution of the problem defined by eqs (11) and (12) is approximated by a boundary interpolation based on the differential operator

$$L := \Delta \Delta(\Delta - c^2 I),$$

since not only the Dirac functional but also its first-order derivatives should be bounded, according to the boundary conditions (12). Now the function

$$\Phi(x) = -(c^2 \| x \|^2 \log(c \| x \|) + 4K_0(c \| x \|) + 4\log(c \| x \|))/8c^4 \pi$$
is the corresponding fundamental solution. The scaling parameter $c$ should again be chosen as earlier.

The interpolation problem defined by the operator (13) supplied with the interpolation conditions (12) taken at the points $x_1, x_2, \ldots, x_N$ has a well-posed variational formulation in the Sobolev space $H^3_0(\Omega_0)$. Now the formulation is based on the closed subspace $W \equiv \{v \in H^3_0(\Omega_0) : \text{grad}v(x_k) = 0, \ k = 1, 2, \ldots, N\}$.

As an illustrative example, we applied the method for the rotating flow with the stream function $\psi(x, y) = x^2 + y^2$ defined on a square with size 1. The number of the boundary points was 128. Table 1 shows how the relative $L_2$-error of the approximate velocity field $u$ depends on the different values of $c$.

<table>
<thead>
<tr>
<th>$c$</th>
<th>50</th>
<th>100</th>
<th>150</th>
<th>200</th>
<th>250</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rel.error (%)</td>
<td>5.53</td>
<td>3.12</td>
<td>2.32</td>
<td>1.95</td>
<td>1.74</td>
</tr>
</tbody>
</table>

Table 1. Different values of the scaling parameter and the relative $L_2$-errors

Acknowledgement: The research was partly supported by the Hungarian Scientific Research Fund (OTKA) under the project T34652.

References


A General Strategy for the Mixed-Dimensional, Symmetric Coupling of FEM and BEM

M. Haas, G. Kuhn

Institute of Applied Mechanics, University of Erlangen-Nuremberg, Egerlandstr. 5, 91058 Erlangen, Germany, haas@ltm.uni-erlangen.de

Keywords: Symmetric Galerkin BEM, SGBEM, FEM/BEM-coupling, kinematic coupling, energy-based coupling, mixed-dimensional coupling

Abstract. When simulating light weight structures, it may be convenient to use formulations of different dimensionality for different domains of the structure, depending on the theory describing its mechanical behavior. As the finite element method (FEM) provides a variety of structural elements it is very efficient for parts of the construction, where assumptions of e.g. beam or shell theory hold. However, for domains with real 3D geometry or 3D states of stress, the boundary element method (BEM) can be the better alternative to FEM, especially when considering the lower complexity for mesh generation, the reduced number of degrees of freedom (DOF) and the higher accuracy. In this paper, a general strategy for coupling the 3D-BEM with FEs of reduced dimension without losing the benefits of the FEM is presented. As an example, the coupling of finite shell elements and 3D boundary elements is discussed.

Introduction

Fig. 1 shows a mixed-dimensional FEM/BEM coupling problem. For the BEM-domain $\Omega^B$ with Dirichlet boundary $\Gamma^u$ and Neumann boundary $\Gamma^t$, the 3D Symmetric Galerkin Boundary Element Method (3D-SGBEM) is used to produce a symmetric and positive definite stiffness matrix. The FE-domain $\Omega^F$ can be modelled with structural elements of reduced dimension such as shell or beam elements. Both domains share the coupling interface $\Gamma^c$. In the following sections, the procedure for the generation of a symmetric and positive definite stiffness formulation for the coupled problem is displayed.

Boundary element domains

The 3D-SGBEM [1, 2] is applied to the 3D domains of the coupling problem. This means that the displacement boundary integral equation (u-BIE) in the sense of Galerkin is evaluated at the Dirichlet boundary $\Gamma^u$ and the
Figure 1: Mixed-dimensional 2D-FEM/3D-BEM coupling problem

corresponding traction-BIE is applied to the Neumann boundary $\Gamma^t$. At
the coupling interfaces, both equations are needed. After a discretization
and regularization process (see e.g. [1]), the following symmetric system
of equations for the computation of the unknown nodal displacements and
tractions can be assembled:

\[
\begin{pmatrix}
-\mathbf{U}_c^c & -\mathbf{U}_u^c & \frac{1}{2}\mathbf{N}_c^c + \mathbf{T}_c^c & \mathbf{T}_t^c \\
-\mathbf{U}_c^u & -\mathbf{U}_u^u & \mathbf{T}_u^u & \mathbf{T}_u^c \\
\frac{1}{2}\mathbf{N}_c^u + \mathbf{U}_c^c & \mathbf{U}_u^c & -\mathbf{T}_u^c & -\mathbf{T}_t^c \\
\mathbf{U}_u^t & \mathbf{U}_u^t & -\mathbf{T}_u^t & -\mathbf{T}_t^t
\end{pmatrix}
\begin{pmatrix}
\mathbf{t}_c \\
\mathbf{t}_u \\
\mathbf{u}_c \\
\mathbf{u}_t
\end{pmatrix}
= \begin{pmatrix}
0 \\
0 \\
\mathbf{N}_c^c \mathbf{t}_c \\
0
\end{pmatrix}
+ \mathbf{R}.
\tag{1}
\]

Indices “u” and “t” refer to the Dirichlet and Neumann boundaries,
respectively, index “c” denotes the coupling interface. The vector $\mathbf{R}$ on
the right hand side gathers all known boundary data, weighted with corre-
sponding integral contributions. The unknown vector $\mathbf{B}_c := \mathbf{N}_c^c \mathbf{t}_c$ contains
work equivalent nodal point forces at $\Gamma^c$. Submatrices $\mathbf{U}_i^j$, $\mathbf{T}_i^j$, $\mathbf{U}_u^t$ and $\mathbf{T}_u^t$
consist of the double boundary integrals arising in a Galerkin formulation
with super- and subscripts denoting the portions of the boundary where
source and field points lie, respectively.

To obtain a stiffness formulation for the BE-domain, all unknown trac-
tions on $\Gamma^u$ and $\Gamma^c$ are eliminated by static condensation and the resulting
system of equations

\[
\begin{pmatrix}
\mathbf{B}_{K_c}^c & \mathbf{B}_{K_c}^u \\
\mathbf{B}_{K_u}^c & \mathbf{B}_{K_u}^u
\end{pmatrix}
\begin{pmatrix}
\mathbf{B}_u \\
\mathbf{B}_c
\end{pmatrix}
= \begin{pmatrix}
\mathbf{B}_f_c \\
\mathbf{0}
\end{pmatrix}
+ \begin{pmatrix}
\mathbf{B}_R_c \\
\mathbf{B}_R
\end{pmatrix}
\tag{2}
\]

is symmetric and positive definite just like the element stiffness matrices of
the FEM [2]. For later convenience all quantities associated with the BE domain are identified by superscript “B” and $B\mathbf{u} := \mathbf{u}_i$ is introduced.

**Finite element domains**

For the FE domain, structural elements can be used, e.g. shell [4] or beam elements. By the principle of virtual work and equilibrium of forces, the FE stiffness formulation

$$
\begin{bmatrix}
F^F \mathbf{K}^F & F^F \mathbf{K}^c \\
F^c \mathbf{K}^F & F^c \mathbf{K}^c
\end{bmatrix}
\begin{bmatrix}
F \mathbf{u} \\
F \mathbf{u}_c
\end{bmatrix}
= 
\begin{bmatrix}
F \mathbf{f} \\
F \mathbf{f}_c
\end{bmatrix}
$$

for the FE domain can be derived. $F\mathbf{u}_c$ are the generalized nodal displacements (i.e. both translational and rotational DOF can occur) at the interface, $F\mathbf{u}$ the ones at the remaining FE nodes. The corresponding vectors on the right hand side are generalized forces which can contain nodal point forces as well as moments, depending on the type of finite elements used.

**Mixed-dimensional FEM/BEM coupling**

In this section, the main features of a general coupling strategy, which is able to deal with the mismatch in DOF at the mixed-dimensional coupling interface, will be discussed.

The first step is to devide the DOF of the FE and the BE domains at the interface in two subsets [2, 3, 6]:

$$
\{F\mathbf{u}_c\} = \{F\mathbf{u}_i\} \cup \{F\mathbf{u}_d\} \quad \text{and} \quad \{B\mathbf{u}_c\} = \{B\mathbf{u}_i\} \cup \{B\mathbf{u}_d\}.
$$

The superscript “i” indicates the subsets of independent DOF, i.e. the DOF which are needed to describe the linear elastic behavior of the system. The other superscript “d” identifies the subsets of dependent DOF which can be expressed in terms of the independent DOF. This can be done by formulating multiple point contraints (MPCs). Generally, the dependent DOF of the FEM domain at the coupling interface can be expressed in terms of the independent DOF of the BEM domain and vice versa by the matrix equations:

$$
F \mathbf{u}_c^d = G_{FB} B \mathbf{u}_i^c, \quad B \mathbf{u}_c^d = G_{BF} F \mathbf{u}_i^c.
$$

In a second step, the vectors $\mathbf{u}$ of dependent DOF and $\mathbf{u}_i$ of independent DOF are defined by

$$
\mathbf{u} = (F\mathbf{u}, F\mathbf{u}_c^i, F\mathbf{u}_c^d, B\mathbf{u}_c, B\mathbf{u}_c^d)^T, \quad \mathbf{u}_i = (F\mathbf{u}, F\mathbf{u}_c^i, B\mathbf{u}_c^d, B\mathbf{u})^T.
$$
Using these definitions, the dependent DOF can be mapped to the independent DOF by:

\[
\begin{bmatrix}
F_u \\
F_u^d \\
B_u^d \\
B_u
\end{bmatrix}
= \begin{bmatrix}
I & 0 & 0 & 0 \\
0 & I & 0 & 0 \\
0 & 0 & G_{FB} & 0 \\
0 & G_{BF} & 0 & 0
\end{bmatrix}
\begin{bmatrix}
F_u \\
F_u^d \\
B_u^d \\
B_u
\end{bmatrix}
\quad \text{or} \quad u = G u_i, \quad (7)
\]

where \(I\) are identity matrices of appropriate size.

After combining the FE (eq (3)) and BE (eq (2)) systems, the associated virtual work expression in terms of vector \(u\) can be written as

\[
\delta u^T (Ku - f - R) = 0. \quad (8)
\]

Under consideration of eq (4), matrix and vectors in eq 8 read:

\[
K = \begin{bmatrix}
F K_F^T & F K_i^F & F K_d^F \\
F K_F^d & F K_i & F K_d^i \\
F K_d^d & 0 & 0
\end{bmatrix}, \quad f = \begin{bmatrix}
F f \\
F f^d \\
0
\end{bmatrix}, \quad R = \begin{bmatrix}
0 \\
0 \\
B R_d \\
B R_i
\end{bmatrix}. \quad (9)
\]

As eqs (5) are not yet taken into account, the above matrix \(K\) is singular.

Both the real and virtual displacements are restricted by eq (5). Thus, the dependent displacements can be eliminated and eq (8) reduces to

\[
\delta u_i^T G^T (K G u_i - f - R) = 0. \quad (10)
\]

As the virtual displacements may be chosen arbitrarily, eq (10) yields the stiffness formulation

\[
K_i u_i = f_i + R_i \quad (11)
\]

with abbreviations

\[
K_i = G^T K G, \quad f_i = G^T f, \quad R_i = G^T R. \quad (12)
\]

The transformation \(G^T f\) of the nodal point force vector ensures that despite the possible discrepancy in the type of force quantities at the coupling interface the conditions of equilibrium can be fulfilled:

\[
F f_i + G_{BF}^T B f_i = 0, \quad G_{FB}^T F f_i + B f_i = 0. \quad (13)
\]
Thus, \( f_i \) reduces to \( f_i = (Ff, 0, 0, 0)^T \). Eq (11) has the main properties of a FEM stiffness formulation, namely symmetry and positive definiteness and can be solved with standard FEM solvers like e.g. Cholesky decomposition or the CG-algorithm.

**Coupling of finite shell elements and 3D boundary elements**

As an example, the coupling of FE element domains and 3D BE domains is considered. Adequate MPC equations can be derived by either analysing the kinematics at the interface [2] or from an energy based formulation [3].

Considering the kinematic constraints at the coupling interface of shell and 3D domain (see fig. 2), one finds that e.g. for BE-node 1

\[
B_u^1 = F_u^1 - \frac{t}{2} F \theta_2^1, \quad B_u^2 = F_u^2 + \frac{t}{2} F \theta_1^1, \quad B_u^3 = F_u^3
\]  

must hold [2]. Formulating such direct kinematic coupling equations for each node at the interface leads to the matrix expression

\[
B_u^d = G_{BF} F_u^c
\]

presenting the MPCs of eq (5) of the previous section.

In the weak coupling case, continuity of the displacements at \( \Gamma^c \) is not demanded explicitly, but rather equivalence in the energy transported across the interface (\( \Pi^{2D} = \Pi^{3D} \)) [5]. The work of the FE nodal forces and moments done on its translational and rotational displacements is

\[
\Pi^{2D} = \int_s \left( M_1^F \theta_1 + N_1^F u_1 + M_2^F \theta_2 + N_2^F u_2 + Q_3^F u_3 \right) ds.
\]

For the coupling interface of the BE domain, the distribution of the stresses derived in the classical theory (bending stresses \( \sigma_2 \) and in-plane shear stresses \( \tau_{12} \) vary linearly, the transverse shear stresses \( \tau_{13} \) quadratically over the shell’s thickness) is used. The work of these stresses performed on the displacements at the interface is
\[ \Pi^{3D} = \int_{A} B_{ij} B_{k} u_{i} dA = \int_{z} \int_{-t/2}^{t/2} (\sigma_{1} B_{u1} + \tau_{12} B_{u2} + \tau_{13} B_{u3}) dz_{3} ds. \] (17)

Comparison of eqs (16) and (17) and integration over the shell’s thickness using the local coordinates of the BEs [3] leads to coupling equations in the form of

\[ F_{u_{i}}^{1} = \frac{1}{6} (B_{u_{i}}^{1} + B_{u_{i}}^{4} + 4 \cdot B_{u_{i}}^{5}) (i = 1, 2), \quad F_{u_{i}}^{3} = \frac{1}{10} (B_{u_{i}}^{3} + B_{u_{i}}^{4} + 8 \cdot B_{u_{i}}^{5}), \]

\[ F_{\theta_{i}}^{1} = -\frac{1}{t} (B_{\theta_{i}}^{1} - B_{\theta_{i}}^{4}), \quad F_{\theta_{i}}^{2} = \frac{1}{t} (B_{\theta_{i}}^{1} - B_{\theta_{i}}^{2}) \] (18)

for FE node 1 as an example. Once again, the coupling equations for the weak formulation can be assembled in matrix form

\[ F u_{c}^{d} = G_{FB} B_{u_{i}}, \] (19)

which is the starting point of eq (5).

**Summary**

In this paper, a generalized coupling strategy for mixed-dimensional FEM/BEM-coupling is presented. The coupling equations are formulated by MPC, which allow to overcome the mismatch in DOF at the interface. The derivation of appropriate equations is illustrated by the coupling of finite shell elements and 3D boundary elements and numerical examples, demonstrating the superiority of the weak coupling strategy, will be presented at the conference.

**References**


