Generic domain decomposition and iterative solvers for 3D BEM problems

F. C. Araújo¹,*, †, K. I. Silva¹, ‡ and J. C. F. Telles², §

¹Department of Civil Engineering, UFOP, CEP: 35400-000, Ouro Preto-MG, Brazil
²Department of Civil Engineering, COPPE/UFRJ, CXP 68506, CEP: 21945-970, Rio de Janeiro-RJ, Brazil

SUMMARY

In the past two decades, considerable improvements concerning integration algorithms and solvers involved in boundary-element formulations have been obtained. First, a great deal of efficient techniques for evaluating singular and quasi-singular boundary-element integrals have been, definitely, established, and second, iterative Krylov solvers have proven to be advantageous when compared to direct ones also including non-Hermitian matrices. The former fact has implied in CPU-time reduction during the assembling of the system of equations and the latter fact in its faster solution. In this paper, a triangle-polar-co-ordinate transformation and the Telles co-ordinate transformation, applied in previous works independently for evaluating singular and quasi-singular integrals, are combined to increase the efficiency of the integration algorithms, and so, to improve the performance of the matrix-assembly routines. In addition, the Jacobi-preconditioned biconjugate gradient (J-BiCG) solver is used to develop a generic substructuring boundary-element algorithm. In this way, it is not only the system solution accelerated but also the computer memory optimized. Discontinuous boundary elements are implemented to simplify the coupling algorithm for a generic number of subregions. Several numerical experiments are carried out to show the performance of the computer code with regard to matrix assembly and the system solving. In the discussion of results, expressed in terms of accuracy and CPU time, advantages and potential applications of the BE code developed are highlighted.

KEY WORDS: singular and quasi-singular integration algorithms; Krylov’s solvers; discontinuous boundary elements; generic BE/BE coupling algorithm

*Correspondence to: F. C. Araújo, Department of Civil Engineering, UFOP, CEP: 35400-000, Ouro Preto-MG, Brazil.
†E-mail: fcelio@em.ufop.br
‡E-mail: kisilva@metalicas.em.ufop.br
§E-mail: telles@coc.ufrj.br

Contract/grant sponsor: Brazilian Research Council, CNPq; contract/grant number: 300.752/2004-0
Contract/grant sponsor: Research Foundation for Minas Gerais, FAPEMIG

Received 15 July 2005
Revised 7 February 2006
Accepted 13 February 2006

Copyright © 2006 John Wiley & Sons, Ltd.
1. INTRODUCTION

Nowadays, boundary element methods (BEM) have been largely applied to solve many different kinds of engineering problems [1–3]. General advantages of this method are its high precision, the response description exclusively in terms of boundary values and the fulfillment of radiation conditions. Therefore, problems defined in domains containing cracks [4, 5] and in open domains as e.g. soil–structure-interaction problems [6–10] have been quite successfully solved through BE analyses. Furthermore, compared to FE models, the mesh generation involved is much simpler and the resulting systems of algebraic equations are considerably smaller. Nevertheless, the presence of singular and quasi-singular integrals, the non-symmetry and non-sparse of the resulting system of equations, and the difficulty in including non-homogeneous characteristics in the analysis might be mentioned as disadvantages of BE methods. In fact, these themes have been intensively addressed in many research works since the early 1980, and the results attained have successfully surpassed most of these challenges. Hence, nowadays, it can be affirmed that BE methods are competitive in many large-order engineering applications [3].

In dealing with non-regularized boundary-integral formulations, not only the accuracy but also the computer run time associated with the computation of singular and nearly-singular integrals plays an important role in the efficiency of BE methods. In general, indirect [7, 11, 12] and direct [13–15] integration schemes, which apply to both stationary and time-dependent problems, have been established for computing the strongly singular integrals (Cauchy principal values). In case of indirect integration algorithms, additional enclosing-element meshes must be considered for modelling half-space domains [7]. Furthermore, the calculation of weakly-singular integrals, also occurring in the implicit computation of the corresponding leading diagonal block-matrices, requires special care. In fact, the computation of weakly-singular and quasi-singular integrals involved in BE methods, though existing in ordinary sense, may be very time-consuming if proper algorithms are not used. Several strategies have been idealized to evaluate them. Those based on the use of integration subelements [16–18] are among the first attempts to compute precisely this kind of integrals. Later, a class of integration methods based on co-ordinate transformations, more efficient and more commonly used nowadays in BE codes, have been introduced [19–23]. A disadvantage of element-subdivision-based methods is that the degree of the highest-degree polynomial exactly integrated through the numerical quadrature is determined by the number of integration points considered in each subelement and not by the total number of integration points per element. As a matter of fact, integration methods based on either subelements or co-ordinate transformation cannot be applied without additional modifications to evaluate correctly Cauchy-principal-value integrals [15]. Nevertheless, the method proposed by Li et al. [19] works well for computing weakly-singular or nearly-weakly-singular integrals.

Another interesting idea towards improving the evaluation of weakly-singular or nearly-singular integrals was proposed by Telles [21], who developed an integration procedure based on a cubic polynomial co-ordinate transformation. According to this procedure, integration parameters are optimized as functions of the position of the source point in relation to the boundary element. As a result, the integration points are displaced towards the source point; yet for enough large source-to-element distance, no co-ordinate transformation is, in effect, observed. This strategy has proven to increase considerably the efficiency of integration algorithms for weakly-singular and nearly-weakly-singular integrals [22].

Indeed, the integration procedures based on co-ordinate transformation do not entirely remove the boundary-layer effect in case of nearly-strongly-singular integrals [24]. It means that a residual error is always expected in the evaluation of this kind of nearly-singular integral for source points very close to the boundary element. However, for quasi-singular source points not so very close to the boundary, at distances numerically quantified later in this paper, co-ordinate-transformation-based integration procedures may furnish high-precision results by using relatively-low-order integration rules. Moreover, other alternatives, as for instance the line integral approach [19, 25, 26], may be additionally implemented to improve even more the integration precision in case of nearly-strongly-singular integrals.

In this paper, an integration algorithm that combines integration schemes based on a triangle polar co-ordinate transformation [19] and the cubic polynomial co-ordinate transformation [21, 22] is proposed. It should be noticed that here, the precise evaluation of quasi-singular integrals is essential also because of the possible presence of discontinuous boundary elements in the discretization models. Furthermore, the study aims at observing the influence of the integration algorithm adopted on a possible reduction of the integration order for usual three-dimensional models, that is, BE models non-containing thin-walled domains.

Another fundamental question concerning BE analyses is the solution of the system of equations, usually very large in practical applications. Here, the research on Krylov’s solvers carried out in the last two decades has, definitely, attested the efficiency of this class of iterative solvers [27–30]. In general, the efficiency of iterative solvers, when compared with direct ones, increases with the system order. Another advantage of this kind of solver is that no transformation matrix is carried out during the solution process. In this way, these solvers are especially adequate for deriving optimized matrix-storage formats, since they make the complete exclusion of zero coefficients possible. In addition, they are also quite suitable for developing codes with a parallel-computing option, since they enable easy problem subdivision by means of generic domain decomposition (substructuring). Many works have pointed out the efficiency of Krylov’s iterative schemes in BE formulations [7, 31–39]. In References [28–31, 40–42], the most important Krylov’s algorithms applied to solve BE systems, such as BiCG, Lanczos, GMRES, CGS e BiCGSTAB, can be found in detail.

The boundary-element substructuring strategy based on iterative solvers has also been addressed in several papers [7–9, 42]. In fact, it is believed that the iterative-solver-based substructuring technique proposed in these works, by the reasons given above, is much more efficient than those based on direct solvers [43–46]. In this paper, an improved version of the generic coupling strategy proposed by the first author in previous works [7–9, 42] is implemented. In this version, discontinuous boundary elements are included. Thus, the simulation of traction discontinuity at inner corners and edges, inevitable for coupling a generic number of subdomains, becomes considerably easier. On the other hand, special care concerning the quasi-singular integrals must be taken into account. As implemented, the models associated with each subregion are treated as if they were in effect independent of one another, so that the algorithm is ready to be implemented on a parallel-computing platform. In fact, it will be shown in this paper that the proposed coupling algorithm may constitute a very attractive strategy, if not the most attractive, for developing parallel BE codes.

Special studies concerning the performance of the different integration procedures implemented in the code are carried out. Details of the coupling algorithm, mainly those associated with the implementation of discontinuous boundary elements, are presented.
In the section of applications, several numerical experiments are carried out to illustrate the performance of the whole code and further improvements are also suggested.

## 2. INTEGRATION PROCEDURES

The coefficients of the $H$ and $G$ matrices in standard direct 3D BE formulations, for scalar or vector problem, result from evaluating, respectively, the following surface integrals over each boundary element $\Gamma_e$:

\[
\begin{align*}
   h_{i,3(q-1)+k} &= \int_{\Gamma_e} p^p_{ik}(\chi; \xi) h_q(\eta_1, \eta_2) \, d\Gamma(\chi) \\
   g_{i,3(q-1)+k} &= \int_{\Gamma_e} u^p_{ik}(\chi; \xi) h_q(\eta_1, \eta_2) \, d\Gamma(\chi)
\end{align*}
\]

where $q$ denotes a local node number, $h_q$ is the interpolation function, $-1 \leq \eta_1 \leq 1$ and $-1 \leq \eta_2 \leq 1$ are the natural co-ordinates, and $i$ and $k$, integer numbers varying from 1 to the number of degrees of freedom per node. The functions $p^p_{ik}$, of order $O(r^{-2})$, and $u^p_{ik}$, of order $O(r^{-1})$, with $r = \|\chi - \xi\|$, are the fundamental tractions and displacements, respectively. $\chi$ is the field point, and $\xi$, the source point.

Several strategies, mainly in the last two decades, have been developed to calculate efficiently the above integrals [16–23, 25, 26]. In this paper, quadratures based on co-ordinate transformations are studied. Namely, polynomial and triangle polar co-ordinate transformations are employed. An improvement achieved by combining these procedures is further suggested, and, by reason of comparison and for the completeness of the study, standard Gauss quadrature is also included.

### 2.1. Polynomial co-ordinate transformation procedures

According to these procedures, a polynomial of $m$th degree is introduced to transform the original abscissas of the Gauss–Legendre quadrature. Thus, for one-dimensional integrals, one has

\[
\int_{-1}^{1} f(\eta) \, d\eta = \int_{\bar{\eta}_i}^{\bar{\eta}_f} f[\eta(\gamma)] J_p(\gamma) \, d\gamma
\]

where $\eta(\gamma)$ is the co-ordinate-transformation polynomial, generically given by

\[
\eta(\gamma) = a_m \gamma^m + a_{m-1} \gamma^{m-1} + \cdots + a_0
\]

and specifically determined by imposing the following generic restrictions:

\[
\begin{align*}
   \eta(\bar{\gamma}_i) &= -1 \\
   \eta(\bar{\gamma}_f) &= 1 \\
   J_p(\gamma = \bar{\gamma}_i) &= \bar{r}_i, \quad \text{with} \quad \eta(\bar{\gamma}_i) = \bar{\eta}_i \\
   \text{higher order derivative conditions at} \quad \bar{\eta}_f
\end{align*}
\]

Copyright © 2006 John Wiley & Sons, Ltd.

In these expressions, $J_p(\gamma)$ is the Jacobian of the polynomial co-ordinate transformation, $\bar{r}_l$, the $l$th integration optimization parameter, and $\bar{\eta}_l$, $-1 \leq \bar{\eta}_l \leq 1$, the abscissa of the $l$th point of $f(\eta)$ where derivative requirements on $\eta(\gamma)$ are imposed. In effect, it is suitable to set $\gamma_l = -1$ and $\gamma_r = 1$, so as to have the same mapped integration interval used for standard Gauss quadrature, i.e. $-1 \leq \gamma \leq 1$. Note that $\bar{\eta}_l$ usually corresponds to a either singular or quasi-singular point, which, particularly for quasi-singular points, should be taken as that abscissa in $[-1, 1]$ for which the real distance $r = \|\chi(\bar{\eta}) - \xi\|$, $\xi$ being the singularity point of $f(\eta)$, is minimal.

In principle, the co-ordinate-transformation polynomial is obtained so that the Jacobian, $J_p(\gamma)$, be cancelled at the singular points ($\bar{r}_l = 0$) and be equal to some non-zero $\bar{r}_l$, determined so as to minimize the integration error, at the quasi-singular points [21, 22]. As the determination of the $\bar{r}_l$ parameters for the regular-integration cases makes polynomial integration procedures more cumbersome, no optimization parameters for quasi-singular points are taken into account in this study.

In case of having $f(\eta)$ a unique singular or quasi-singular point at $\bar{\eta} = -1$, a family of co-ordinate-transformation polynomials given by

$$\eta(\gamma) = 2^{1-m}(\gamma + 1)^m - 1$$

(6)

satisfying the conditions

$$\eta(-1) = -1$$

$$\eta(1) = 1$$

(7)

$$J_p(-1) = 0 \quad \text{if} \quad m \geq 2$$

$$\eta''(-1) = \eta'''(-1) = \cdots = \eta^{(m-1)}(-1) = 0 \quad \text{if} \quad m \geq 3$$

is obtained. Particularly for $m = 3$, the polynomial described by Equation (6) corresponds to the Telles’ cubic transformation for the conditions given by relations (7) (see References [21, 22]). In fact, the cubic Telles’ transformation [21] is more general since it was established for a generic abscissa $\bar{\eta}$ in $-1 \leq \bar{\eta} \leq 1$. In Figure 1, the effect of the polynomial co-ordinate transformation on the integration domain is visualized for different polynomial degrees, $m$. As observed, the higher the $m$ value is, the more shifted towards the point, at $\bar{\eta} = -1$, the integration points are. On the other hand, for $m = 1$, standard Gauss abscissas, associated with a non-transformed integration domain, take place.

Thus, by using polynomial co-ordinate transformation, the following formula for evaluating integrals over 3D boundary elements can be used:

$$\int_{\Gamma_e} f_{ik}^\alpha(\xi; \eta_1, \eta_2) h_\eta(\eta_1, \eta_2) d\Gamma_e$$

$$= \int_{-1}^{1} \int_{-1}^{1} f_{ik}^\alpha(\eta_1, \eta_2; \xi) h_\eta(\eta_1, \eta_2) J_p(\gamma_1, \gamma_2) d\eta_1 d\eta_2$$

$$= \int_{-1}^{1} \int_{-1}^{1} f_{ik}^\alpha(\gamma_1, \gamma_2; \xi) h_\eta(\gamma_1, \gamma_2) J_p(\gamma_1, \gamma_2) d\gamma_1 d\gamma_2$$

(8)
where \( f^*_{ik} \) denotes either fundamental tractions or displacements, and the polynomial transformation applied for \( \gamma_1 \) and \( \gamma_2 \) may not be the same.

2.2. Triangle polar co-ordinate transformation procedures

Integration procedures based on triangle polar co-ordinate transformation [19] consist of decomposing the integration domains (3D boundary elements) into a convenient number of triangular integration subdomains, which are in turn mapped onto square integration domains in the respective intrinsic co-ordinate system (see Figure 2), over which the integrals are performed. Then, as the final integration domain is square, standard Gauss quadrature can be applied. Thereby, the boundary integrals shown in Equations (1) and (2) are calculated by

\[
\int_{\Gamma_e} f^*_{ik}(\xi; \zeta) h_q(\eta_1, \eta_2) \, d\Gamma(\chi) = \int_{-1}^{1} \int_{-1}^{1} f^*_{ik}[(\eta_1, \eta_2); \xi] h_q(\eta_1, \eta_2) J^{(e)}(\eta_1, \eta_2) \, d\eta_1 \, d\eta_2
\]

\[
= \frac{1}{4} \sum_{g=1}^{n_t d} \int_{-1}^{1} \int_{-1}^{1} f^*_{ik}[(\xi_1, \xi_2); \xi] h_q(\xi_1, \xi_2) J^{(e)}(\xi_1, \xi_2)(1 + \xi_1) A_g \, d\xi_1 \, d\xi_2 \tag{9}
\]
where ntd (usually 2 or 3) denotes the number of triangular domains considered, $A_g$ is the area of the $g$th triangle, and $f_{ik}^*$, the fundamental kernels.

### 2.3. Combined co-ordinate-transformation procedures

In this section, a new attempt to improve integration procedures is made by combining the schemes previously discussed. Namely, the polynomial co-ordinate transformation given by Equation (6), which displaces the integration points towards the position $\zeta_1 = -1$, is applied in the $\zeta_1$ direction associated with the triangle-polar-co-ordinate transformation. Hence, applying this strategy to Equation (9)

$$\int_{\Gamma_1} f_{ik}^*(\chi; \zeta) h_q(\eta_1, \eta_2) \, d\Gamma(\chi)$$

$$= \frac{1}{4} \sum_{g=1}^{ntd} \int_{-1}^{1} \int_{-1}^{1} f_{ik}^*[\zeta_1(\gamma), \zeta_2]; \zeta] h_q[\zeta_1(\gamma), \zeta_2] J^{(e)}[\zeta_1(\gamma), \zeta_2]$$

$$\times [1 + \zeta_1(\gamma)] A_g J_p(\gamma) \, d\gamma \, d\zeta_2$$

(10)

As observed in Figure 2, after applying the triangle-polar-co-ordinate transformation to the boundary integrals, the position of the singular pole is always at $\zeta_1 = -1$. Thus, no other polynomial transformation for $\zeta_1$, i.e. associated with other singular-point positions in this direction, is necessary. Furthermore, as the integration is regular in the $\zeta_2$ direction, no transformation is necessary for this intrinsic co-ordinate as well. Additional comments are perhaps now due; even though analytical or semi-analytical integration procedures [47, 48] can be more efficient for the weakly-singular integrals of the diagonal sub-matrices of matrix $G$, the present authors have adopted the combined co-ordinate-transformation technique for such integrals just to keep a unified procedure for the quadrature routines. The sub-matrices of
2.4. Numerical tests

To verify the precision of the numerical quadratures discussed above, case studies involving the evaluation of weakly-singular, nearly-weakly-singular, and nearly-strongly-singular surface integrals are considered. Namely, the following boundary integrals over the eight-node element shown in Figure 3 are calculated:

\[ h_{33} = \int_{\Gamma_e} p_{33}^*(\chi; \zeta) h_1(\eta_1, \eta_2) d\Gamma(\chi) \]  
\[ g_{33} = \int_{\Gamma_e} u_{33}^*(\chi; \zeta) h_1(\eta_1, \eta_2) d\Gamma(\chi) \]

where \( u_{33}^* \) and \( p_{33}^* \) are components of the elastostatic fundamental displacements and tractions, respectively, and \( h_1(\eta_1, \eta_2) \) is the shape function associated with the 1st node of the element.

The integrals are evaluated for different \( d/a \) relations, where \( d \) is the distance from the source point to the nearest element node (in this case study, node 1), and \( a \) is the length of the largest side of the element (see Figure 3). In Figures 4 and 5, sample results for
$d/a = 0.01$ in terms of the relative error are presented as a function of the integration order $n$ ($n \times n$ integration points). The integration procedures are identified by the variable icpt, which is set at 0, 1, 2, and 3 for defining, respectively, standard Gauss (icpt = 0), Gauss modified by the triangle-polar-co-ordinate transformation (icpt = 1), Gauss modified by a cubic polynomial co-ordinate transformation (icpt = 2), and Gauss modified by the combined co-ordinate transformation involving triangle-polar-co-ordinate and the cubic polynomial co-ordinate transformation (icpt = 3). The following expression was considered for measuring the relative integration error:

$$
\varepsilon_{rel} = \frac{|r_i - r_e|}{r_e}
$$

where $r_i$ is the result obtained with $i \times i$ integration points, and $r_e$, a highly precise integration result, calculated through procedure icpt = 1 using $6 \times 6$ integration subelements and $20 \times 20$ points per subelement.

As observed in Figure 4, the triangle-polar-co-ordinate-transformation-based procedure (icpt = 1) works very well for evaluating the weakly-singular integral ($d/a = 0$). In this case, acceptable precision is attained already with $4 \times 4$ integration points. This fact is a consequence of the reduction of the singularity order provided by the co-ordinate transformation [19]. But for evaluating nearly-weakly-singular integrals, this procedure becomes less efficient than procedures icpt = 2 (based on the cubic polynomial transformation) and icpt = 3 (based on the combined co-ordinate transformation). Thus, in case of nearly-weakly-singular integrals, shifting the integration points towards the singularity position is a more efficient strategy. Good-precision results for evaluating $g_{33}$ are obtained using procedure icpt = 2 with $6 \times 6$ or more integration points for all $d/a$ relations. Using procedure icpt = 3, enough precision is reached already with integration order $n = 5$ for all $d/a$. For the nearly-strongly-singular integrals, procedure icpt = 2 is also in general somewhat more efficient than procedure icpt = 1, but considerably less efficient than procedure icpt = 3, which produces acceptable results for all $d/a$ observed with integration order $n \geq 8$. Notice that the source points adopted in the
numerical experiments involving nearly-singular integrals are in effect very close to the boundary element \((0.005 \leq d/a \leq 0.05)\). The combined co-ordinate-transformation procedure, icpt = 3, has shown to be robust and has been the most efficient in the tests. As can also be verified in Figures 4 and 5, the standard Gauss procedure (icpt = 0) furnishes in general unacceptable precision for evaluating either weakly-singular or nearly-singular integrals.

3. KRYLOV’S SOLVERS AND THE COUPLING STRATEGY

A noticeable characteristic of iterative solvers is the non-transformation of the coefficient matrix during the solving phase. Thus, iterative solvers have enabled to develop efficient matrix-storage formats aiming at minimizing the necessary computer memory. Furthermore, the non-processing of zeroes and the high convergence rate of some iterative solvers may diminish considerably the processing time. In finite-element analyses, for instance, Krylov’s iterative solvers adapted for element-by-element or edge-based data structures have been efficiently applied to solve computational fluid dynamics problems [49–52]. In fact, the most-commonly thought alternatives for analysing large-scale models combine Krylov’s solvers and efficient storage formats. In this way, codes for operating in parallel-computing platforms may be suitably developed as well.

For systems of equations originated from boundary element formulations, several alternative Krylov’s solvers have been widely tested [32–35, 42]. Among them, biconjugate gradient schemes (BiCG), a Petrov–Galerkin-type iterative solver based on a two-term recurrence formula, have shown a general good performance. Hybrid schemes such as CGS [40] and Bi-CGSTAB [41], and minimal residual approaches as GMRES [28] have also been applied in BE analyses. In fact, the latter approaches (hybrid and minimal residual ones) may converge faster than BiCG schemes in some cases, but they present, in general, a more irregular behaviour, being less efficient in many practical applications. For instance, GMRES methods usually present faster convergence rate than BiCG, but non-rarely do stall. In this study, the Jacobi-preconditioned biconjugate gradient solver (J-BiCG) will be the only solver used along with a BE multi-zone algorithm. Details of its formulation are given in References [31, 32, 42].

In FE formulations, edge-based data structures are a consequence of the coefficient superposition associated with nodes common to different elements. In BE formulations, contrarily, coefficients originated from a node common to different subregions are allocated in different positions in the system matrix. For this reason, the subregion-by-subregion storing format used here is, a priori, an optimized data structure for computing the matrix–vector products involved in the iterative Krylov’s solvers.

Explicitly, the coupled system of equations is like the one shown in Figure 6 for four generic subregions. In this system, \(H_{ij}\) and \(G_{ij}\) denote usual BE matrices obtained for source points pertaining to subregion \(\Omega_i\) and associated, respectively, with the boundary values \(u_{ij}\) and \(p_{ij}\) at \(\Gamma_{ij}\). Notice that \(\Gamma_{ij}\) with \(i \neq j\) is the interface between \(\Omega_i\) and \(\Omega_j\), and \(\Gamma_{ii}\) is the outer boundary of \(\Omega_i\). Generically (for \(n_s\) subregions), the coupled system is given by

\[
\sum_{m=1}^{i-1} (H_{im}u_{mi} - G_{im}p_{im}) + A_i x_i + \sum_{m=i+1}^{n} (H_{im}u_{im} + G_{im}p_{mi}) = B_i y_i, \quad i = 1, n_s
\]  

Indeed, the system of equations (14) is not explicitly assembled. A subregion-by-subregion data structure is used instead for storing its coefficients. In other words, no block of zero coefficients is stored.
In fact, it is supposed in the data-structure model shown in Figure 6 that discontinuous boundary elements are considered for simulating traction discontinuity at inner edges and corners. In this way, a node of a certain subregion may be coupled, exclusively, with only another node of the whole BE model (see Figure 7). Therefore, only compatibility and equilibrium conditions at the interfaces, given by

\[ u_{ij} = u_{ji} \]
\[ p_{ij} = -p_{ji} \]  

have to be imposed for coupling the BE models considered. In relations (15), \( u_{ij} \) denotes the interface-value vector containing potentials or displacements of nodes pertaining to \( \Omega_i \) and placed at \( \Gamma_{ij} \), and \( u_{ji} \), the vector containing potentials or displacements of nodes pertaining to \( \Omega_j \) and placed at \( \Gamma_{ij} \). The same index meaning is used for flux and traction interface values. For elasticity problems for instance, the coupling conditions given in (15) must not be necessarily applied in all directions. Thus, one direction may be coupled while the other is subjected to prescribed boundary conditions. Of course, inclined interfaces may also occur in this situation. They are simulated in the same way as inclined supports, i.e. by referring the nodal displacement and traction vectors at that surface to a local co-ordinate system.
In the computer code, a discontinuous boundary element is generated automatically by displacing all nodes of a continuous boundary element towards its interior. A length $d$, measured along the intrinsic co-ordinate axes, uniform for the whole BE model, is used for calculating the position of its functional nodes (displaced nodes where the field variables are evaluated). In Figure 8, the interpolation function associated with node 1 of the eight-node serendipity discontinuous boundary element, $h^{(d)}_1$, is shown. As known, at a discontinuous boundary element, the geometry remains approximated through the shape functions of the corresponding continuous element (obtained for the non-displaced geometrical element nodes). Both types of boundary elements (continuous and discontinuous) may be simultaneously present at a model to be analysed with the developed code. In fact, since they bring about a considerable increase in the system order, discontinuous boundary elements should be used only when strictly necessary.

Other interesting issue regarding the use of discontinuous boundary elements is the evaluation of the singular and unavoidable nearly-singular integrals. As commented before, if proper integration procedures are not applied, the precise evaluation of quasi-singular integrals, mainly for very small $d$ values, may also be very time-consuming. In this study, the numerical
quadratures previously detailed in Section 2 are employed for the direct evaluation of weakly-
singular and nearly-singular integrals. Cauchy-principal-value integrals are calculated indirectly
through the criterion of constant potential or rigid body displacement for scalar and vector
problems, respectively. Note, that if no analysis option with discontinuous boundary element
were available in the code, a much more complex coupled system would arise [42].

4. APPLICATIONS

In what follows, 3D potential and elastostatic problems are simulated aiming at observing the
potentialities and efficiency of the proposed strategies. Case studies involving the coupling
of several complex BE models, possibly associated with specially ill-conditioned systems of
algebraic equations, are specially highlighted. Curves showing the accuracy and CPU-time
measurements for assembling and solving the system of equations are presented. The solutions
obtained with the code are compared, whenever possible, with ones either available in the
technical literature or calculated using commercial computer packages (ANSYS and IMSL
routines). The tolerance number adopted for stopping the iterative solver was $\varepsilon = 10^{-5}$. All
analyses were carried out at a Pentium III PC with 1 GHz processor clock, 768 MBytes RAM
and the code was developed using the COMPAQ VISUAL FORTRAN compiler, professional
edition 6.5.0.

4.1. Case study 1: 3D heat conduction problem

In this case study, the 3D heat conduction problem sketched in Figure 9 is analysed. Eight
subregions are used to model the whole domain, and two different BE models per subregion
are adopted. In model I, each subregion is discretized with 54 boundary elements (9 eight-node
serendipty boundary elements per subregion face), and in model II, with 96 (16 boundary
elements per subregion face), originating systems with 2528 and 4424 equations, respectively.
Temperature is prescribed at the ends of the specimen, and flux, over the rest of its boundary

Figure 9. Domain decomposition in eight subregions.
(see Figure 9). Different situations are simulated by assuming different thermal conductivities per subregion and different $d$ values for generating the discontinuous elements. Namely, $d = 0.01$, 0.05 and 0.1. Also, the following relative conductivities ($k_i$ denoting the conductivity of the subdomain $\Omega_i$) are adopted: $k_1 = k$, $k_1 = 10^3 k$, $k_1 = 10^6 k$, $k_1 = 10^9 k$, $k_1 = 10^{10} k$, $k_1 = 10^{11} k$, $k_1 = 10^{12} k$, and $k_2 = k_3 = k_4 = k_5 = k_6 = k_7 = k_8 = k$.

The temperature at the central point of $\Omega_5$ is chosen for comparing the solutions obtained with the developed program, NAESY, and the ANSYS code (see Figure 10). In Figures 11–13, relative CPU-time measurements and number of iterations are furnished. It is mentioned that the assembly time for the different integration procedures was obtained using $8 \times 8$ integration points. In Figure 14, the condition number is plotted against the relative conductivity, $k_1/k$, and finally, in Figure 15, the relative error found in the evaluation of the flux at the central point of $\Omega_5$ for $k_1/k = 1$ is shown as a function of the integration order varying from 3 to 20. The conductivity relation $k_1/k = 1$ is chosen because for that an analytical solution is known.

In this problem, the Telles cubic co-ordinate transformation and the triangle polar co-ordinate transformation also produced satisfactory precision, for all $d$ values adopted, with small integration order (error in general inferior to 1.0% by the evaluation of the flux at the central point of $\Omega_5$). But, the former procedure requires somewhat less CPU time for assembling the system of equations than the latter. As seen in Figure 11, which presents the assembly CPU time for models I and II of case study 1, the Telles procedure requires about 80% of the CPU time for the combined procedure. It should still be noticed that no considerable difference on the assembly CPU time is observed between the combined and the triangle-polar-co-ordinate transformation (Figure 11).

For this problem, the performance of the J-BiCG solver and of the generic coupling algorithm, is assessed in terms of CPU time and number of iterations (nit). As Figures 12(a) and (b) show, the J-BiCG solver is about 3–5 times faster than the IMSL-DLSARG one for BE model I.
with 2528 equations, and about 5–8 times faster than the IMSL-DLSARG one for BE model 2, with 4424 equations. The number of iterations obtained varies from 8 to 10% of the system order for model I (see Figure 13(a)), and from 5 to 8% of the system order for model II (see Figure 13(b)). Based on Figures 12 and 13, one sees that the performance is not strongly sensitive to the increase of the relative conductivity. However, for \( d = 0.01 \), the performance...
decayed in comparison to that for $d = 0.05$ and $d = 0.1$. In fact, despite converge, an accuracy decrease is expected if the condition number increases and the tolerance number, $\zeta$, is not accordingly reduced.

In Figure 10(a), results calculated with the NAESY code using model I, different $d$ values and different integration procedures are compared to the ANSYS ones. In Figure 10(b), NAESY results are compared to the ANSYS solutions and the accuracy of the J-BiCG and
IMSL-DLSARG are also observed. As can be seen, the ANSYS results are no longer correct for \( k_1/k \geq 10^{11} \), when the system becomes very ill-conditioned (condition number \( \kappa \geq 10^{12} \)).

4.2. Case study 2: Foundation–soil interaction

Here, a static analysis of the foundation–soil system shown in Figure 16 is carried out. The BE model for the soil has 300 boundary elements and 80 enclosing elements, and for the block foundation, 264 elements, so that the coupled system has a total of 5229 equations. In this case, no discontinuous boundary elements are necessary, and physically different situations are simulated by assuming the following rigidity ratios: \( E_f/E_s = 1, 50, 100, 500, 1000, \) and 10 000, where \( E_f \) and \( E_s \) are the elasticity moduli of the foundation and soil, respectively.

Different integration orders are considered. The rigid-foundation settlement, obtained, e.g. for \( E_f/E_s = 10000 \), is shown in Figure 17(a). That value is compared to those obtained with semi-analytical formulas given by Whitman and Richart [53] and Sieffert and Cevaer [54]. In Figure 17(b), the complete displacement profile of the soil surface at the symmetry plane is given for all \( E_f/E_s \) relations. The bulb of the maximal principal stress, \( \sigma_1 \), for rigidity ratio \( E_f/E_s = 100 \), is plotted in Figure 17(c). In Figures 18(a) and (b), performance data (CPU-time measurement ratios) are furnished.

In this application, only integration procedures icpt = 1 and 3, with different integration orders, are considered. As seen, the foundation settlement calculated with the developed code, NAESY, is confined by the two estimates obtained using the procedures found in References [53, 54] (see Figure 17(a)). Additionally, the displacement profiles at the \( y = 0 \) plane and the bulb of the main stress \( \sigma_1 \) have the expected aspects (see Figures 17(b) and (c)). Note that the solution convergence is already verified with \( 3 \times 3 \) integration points in both integration procedures tested. In Figures 18(a) and (b), performance data for the various integration orders adopted are plotted as functions of the rigidity relation, \( E_f/E_s \). As observed, the J-BiCG solver is about 2–4 times faster than IMSL-DLSARG (see Figure 18(a)), and the number of iterations varies from 4 to 10% of the system order \( n = 5229 \) (Figure 18(b)).
No considerable difference on the assembly CPU time is observed between the combined and the triangle-polar-co-ordinate transformation (Figure 18(c)).

4.3. Case study 3: Foundation–soil–foundation interaction

In this case study, the static interaction through the soil between two surface foundations is analysed. The BE model shown in Figure 19, having 600 boundary elements and 128 enclosing elements for modelling the soil, and 264 elements for modelling the foundation (the same foundation mesh used in the case study 2 above) is considered. The resulting system order is 10 383 and again the rigidity ratios $E_I/E_S = 1, 50, 100, 500, 1000,$ and $10000$ are taken into account. Integration orders varying from 3 to 8 are adopted.
Figure 17. (a) Rigid-foundation settlement; (b) displacement profile at $y=0$ plane (NAESY-J-BiCG, icpt = 3, 8 x 8 integ. points); and (c) max. prin. stress, $\sigma_1$, in N/mm$^2$, for $E_f/E_s = 100$. 

Copyright © 2006 John Wiley & Sons, Ltd.  
The displacement profile of the soil surface for the different rigidity ratios adopted is given in Figure 20(a), and the bulb of the main stress \( \sigma_1 \) for \( E_t/E_s = 100 \) is shown in Figure 20(b). This problem was solved using \( 3 \times 3, 4 \times 4, 5 \times 5, 6 \times 6, \) and \( 8 \times 8 \) integration points and solution convergence was reached already with \( 3 \times 3 \) integration points. The CPU-time ratio and number of iterations (nit), scaled by the system order \( (n) \), obtained using the J-BiCG solver are plotted against the rigidity ratio \( E_t/E_s \) in Figures 21(a) and (b), respectively.

Here, no comparisons with semi-analytical solutions are presented. But the good accuracy observed in the previous analyses encouraged the authors to publish the results obtained with the NAESY code in this case study. Furthermore, the expected aspect of the displacement profile of the soil surface at the \( y = 0 \) plane (see Figure 20(a)) and of the bulb of the main stress \( \sigma_1 \) (see Figure 20(b)) appears to be entirely consistent. Again, only integration procedures icpt = 1 and 3, with different integration orders, are applied. As a matter of fact, the solution already converged with \( 3 \times 3 \) integration points. In this case study, it was not possible to solve the corresponding system of equations with the IMSL-DLSARG solver since the system
order \((n = 10383)\) exceeded the maximal allowed value. Therefore, the only solver used was the J-BiCG. In Figures 21(a) and (b), CPU time measurements and the number of iterations are, respectively, plotted for the different integration orders adopted as a function of \(E_f/E_s\). Note that for \(3 \times 3\) integration the solver performance decays. Then it is expected that for this low integration order the system conditioning worsens. For integration orders superior to 4, no considerable performance change is observed (see Figure 21(b)).

5. CONCLUSIONS

The performance of the procedures proposed in this paper, \textit{a priori} concerned with the increase of efficiency of BE codes, has been observed by carrying out several, somewhat burdensome, three-dimensional numerical simulations. As verified, discontinuous boundary elements make the generic coupling algorithm considerably easier, because traction discontinuities may be simulated by applying only equilibrium and compatibility conditions (Equations (15)). Nevertheless, the evaluation of the quasi-singular integrals is inevitable. In this respect, the performance study of

---

Figure 19. BE model for the foundation–soil–foundation system: (a) side view; and (b) upper view.
Figure 20. (a) Displacement profile at $y = 0$ plane (NAESY-J-BiCG, icpt = 3, $8 \times 8$ integ. points); and (b) maximal principal stress, $\sigma_1$ (in N/mm$^2$), for $E_f/E_s = 100$.

Figure 21. (a) J-BiCG solver/assembly CPU-time ratio; and (b) num. of iterations (nit)/system order ($n$).
the integration procedures considered in Section 2 has shown that the combined co-ordinate-transformation procedure proposed (icpt = 3) is the most robust of the procedures tested and can be satisfactorily applied for acceptable integration orders \( n \leq 8 \).

Concerning the J-BiCG solver, no considerable performance decay has been observed as the relative conductivity increases (see Figures 13, 18(b), 21(b)). However, a worse performance of the solver is expected, for very small \( d \) values (see Figure 13), associated with quasi-singular systems. Indeed the J-BiCG solver has proven to be reliable and fast, converging even for the very ill-conditioned systems as for instance those associated with either high conductivities or rigid foundations or very small node shifting in discontinuous elements. In addition, the computer-memory saving is also significant. As seen, for instance, from case study 1, the sparsity of the global matrices for model I and II is about 78%, and for the BE models adopted in case studies 2 and 3, about 28 and 43%, respectively.

CPU time and memory reduction, either during assembling or solving the system of equations, is the chief goal of the coupling algorithm proposed. In fact, the solving CPU time can be further reduced by structuring the matrix–vector product involved in the iterative solver. Then, some of the conditional commands occurring in the corresponding multiplication routine can be removed. Finally, it should be observed that the coupling algorithm proposed can easily be implemented in parallel-computing platforms, being therefore promising for analysing large-scale models.

ACKNOWLEDGEMENTS

The authors would like to acknowledge the Brazilian Research Council, CNPq, and the Research Foundation for Minas Gerais, FAPEMIG, which have funded this work.

REFERENCES


