Iterative coupling algorithms for large multi-domain problems with the boundary element method

Received <day> <Month>, <year>; | Revised <day> <Month>, <year>; | Accepted <day> <Month>, <year>

DOI: xxx/xxxx

ARTICLE TYPE

Iterative coupling algorithms for large multi-domain problems with the Boundary Element Method

Bin Wang[1] | Yin Feng*[1] | Sandra Pieraccini[2] | Stefano Scialò[3] | Corrado Fidelibus[4]

[1]Department of Petroleum Engineering, University of Louisiana at Lafayette, Lousiana, USA
[2]Dipartimento di Ingegneria Meccanica e Aerospaziale, Politecnico di Torino, Italy
[3]Dipartimento di Scienze Matematiche, Politecnico di Torino, Italy
[4]Dipartimento di Ingegneria Civile, Ambientale, del Territorio, Edile e di Chimica, Politecnico di Bari, Italy

Correspondence
*Yin Feng, Department of Petroleum Engineering, University of Louisiana at Lafayette. Email: yin.feng@louisiana.edu

Present Address
Corrado Fidelibus, Dipartimento di Ingegneria dell'Innovazione, Università del Salento, Lecce, Italy

Abstract

A new Parallel Robin-Robin (P-RR) adaptive Iterative Coupling (IC) algorithm with dynamic relaxation parameters is proposed for the Boundary Element Method (BEM), and relaxation parameters are derived for other existing IC algorithms. The performances of the new algorithm and of the modified existing algorithms are investigated in terms of convergence properties with respect to the number of sub-domains, mesh density, interface mesh conformity and BEM element types. Results show that the number of sub-domains and the refinement level of the mesh are the two dominant factors affecting the performances of the considered algorithms. The proposed P-RR algorithm shows the best overall convergence behavior for the tested large problems, thanks to its effectiveness in handling complex boundary conditions and large number of sub-domains, thus resulting very promising for efficient parallel BEM computing and large coupling problems.
Source code is available at https://github.com/BinWang0213/PyBEM2D.

KEYWORDS:
Iterative coupling method; Boundary Element Method; Adaptive relaxation; Domain Decomposition Method

1 | INTRODUCTION

The Boundary Element Method (BEM)[1,2] has been successfully applied to many engineering fields, such as fracture mechanics, acoustics, electromagnetics, heat transfer and fluid flow problems, because of some advantageous peculiarities, like boundary-only discretization and efficient handling of both complex geometries and infinite domains[3,4,5,6,7,8,9,10,11,12,13,14]. However, the systems of equations which arise from

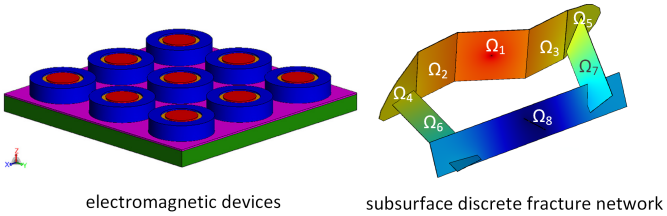


FIGURE 1 Typical multi-domain problems

its application are non-symmetric and dense, thus the method is not practical in solving for large engineering problems, unless specific strategies are adopted to reduce the computational demand. The fast multi-pole method and hierarchical matrices were applied to expedite matrix-assembly and matrix-vector-product operations and, for problems with thousands of Degrees Of Freedom (DOFs), the computational time was reduced to one tenth[15,16]. Other strategies rely upon high performance parallel computing capabilities of modern supercomputers, such as Graphics Processing Units (GPUs)[17]. For example, Takahashi and Hamada[18] used GPUs to parallelize matrix-vector-product operations for the solution of a three-dimensional Helmholtz problem, and achieved an improvement of 6 to 23 times in computational speed with respect to the performance obtained by using a sequential BEM algorithm. Copper, Bardhan and Barba[19] implemented a multi-pole method in an accelerated BEM code for biomolecular electrostatic analyses. Torky and Rashed[20] developed a commercial BEM code for elastostatic problems, based on GPU computing, in which matrix-assembling operations and the direct linear algebra solver are both accelerated; for a very large model, the authors obtained a reduction of the computational time from several hours to few minutes.

For large problems, difficulties may emerge when using BEM to account for multi-domain problems, as, for example, the computation of the electromagnetic field in industrial objects composed of materials having different properties[21], or the fluid flow regime in discrete fracture networks[22,23]. In fact, for the solution of this kind of problems, the direct coupling, i.e. the assembly of a large system of equations including the compatibility conditions at the interfaces among adjacent domains, may be too computationally demanding. In this respect, reasonable alternatives are Iterative Coupling (IC) algorithms, where, instead of one global system, several small sub-problems are iteratively solved with conditions at the interfaces that are updated at each iteration according to specific criteria, until obtaining a solution that fits the compatibility conditions at the interfaces[25,26,27,28,24,29,30,31,32,33,34,35].

reference	algorithm	coupling	relaxation	# s.d.	mesh
[25]	P-DD, P-NN, P-DN	BEM-BEM	NA	2-4	C
[26,27]	P-NN, P-DN, P-DN, P-RR	BEM-FEM, BEM-BEM	NA/A	2-4	C
[28]	S-DN	BEM-FEM	A	2	C
[29]	S-DN	BEM-FEM	A	2-3	NC
[31]	S-DN, P-NN, P-DD	BEM-FEM	A	2	C
[38,39,40,41,42,43]	P-RR	FEM-FEM	A	2-100+	NC
herein	P-DD, P-NN, P-RR, S-DN	BEM-BEM	A	3-45	NC

TABLE 1 Summary of IC algorithms; A/NA: adaptive/non-adaptive, # s.d.: number of sub-domains, C/NC: conforming/non-conforming mesh

In literature, several contributions were presented regarding the application of combined IC algorithms and Domain Decomposition Methods (DDM) within the framework of BEM. Kamiya, Iwase and Kita[25] presented three different IC+DDM algorithms for two-dimensional potential problems and analyzed the related convergence responses. Similarly, Elleithy et al.[26] and Elleithy and Tanaka[27] investigated the effect of the relaxation parameter on the convergence of both parallel and sequential IC algorithms for multi-domain potential and elastic problems. In these two works, the IC was used for multi-domain problems in which different discretization strategies, namely the BEM and the Finite Element Method (FEM), were applied to different domains to profit of the different approximation properties. It was also found that the convergence of these algorithms is very sensitive to the selection of the relaxation parameter, a circumstance that may limit their applicability.

In order to increase the efficiency of IC algorithms, Adaptive Iterative Coupling (AIC) algorithms have been recently proposed in the context of BEM and also of the Finite Element Method (FEM), in which the relaxation parameter is dynamically determined at each iteration step. Lin et al.[28] developed an optimization procedure with an adaptive relaxation parameter for a sequential Dirichlet-Neumann (S-DN) IC algorithm, in which Dirichlet and Neumann type boundary conditions at the interfaces are sequentially updated with iterations. Soares[29] and Soares and Godinho[30] extended the algorithm to transient heat conduction problems, where a specific handling of the non-conforming BEM and FEM meshes at the interfaces was introduced. Francois et al.[31] developed a new AIC algorithm based on the Aitken's Δ^2 method, that was applied to a three-dimensional transient elastodynamic problem, where sequential Dirichlet-Neumann algorithm and parallel Neumann-Neumann (P-NN) and Dirichlet-Dirichlet (P-DD) algorithms were considered. In parallel algorithms, the resolution of the problems at the interfaces is performed simultaneously at each iteration, thus gaining a reduction of the computational time.

An innovative IC algorithm has been recently proposed for the fluid flow in discrete fracture networks with FEM-based discretization techniques. A quadratic cost functional, including interface conditions, is constrained by partial differential equations on each sub-domain and minimized[38,39,40,41,42,43]. In order to ensure well posedness of the sub-domain problems, Robin-type interface conditions are used. The minimization is pursued by using a gradient-based approach.

Table 1 reports a list of the IC algorithms available in literature. It is worth to note that, despite variety and number, most of them concern BEM-FEM coupling and are limited to a small number of sub-domains and static relaxation parameters. Thus, room is left for new algorithms in large multi-domain problems based on BEM-BEM coupling.

In this note, a new AIC algorithm, named *Parallel Robin-Robin* (P-RR), is proposed, having the following features: Robin-type conditions are utilized at the interfaces among sub-domains and a relaxation parameter is dynamically chosen at each iteration, following a procedure similar to the one proposed by Lin et al.[28], to reduce the iteration steps. As shown in the following, P-RR has good convergence properties and is expected to properly work for problems with a large number of domains and non-conforming meshes. In addition, the Parallel Dirichlet-Dirichlet (P-DD) and Parallel Neumann-Neumann (P-NN) algorithms are revisited and endowed with adaptive relaxation parameters.

The content of this note is organized as follows: in Section 2, BEM is briefly recalled and the relevant notation is introduced; in Section 3, the new P-RR algorithm is proposed, together with the update of P-DD and P-NN algorithms; finally, in Section 4 numerical examples to test and compare the response of the different algorithms in multi-domain problems of increasing size and complexity are discussed. The effects on the convergence of discretization-related factors, such as mesh density, interface conformity and BEM element types, are also investigated.

2 | BEM EQUATIONS

In two dimensions, with reference to a coordinate system $\mathbf{x} = \{x_1, x_2\}$, the steady-state potential p in an isotropic and homogeneous domain Ω, enclosed by a boundary Γ, is governed by the Laplace equation:

$$\nabla^2 p(\mathbf{x}) = 0, \qquad \mathbf{x} \in \Omega. \tag{1}$$

The equation is associated to Dirichlet and/or Neumann boundary conditions on Γ_D and Γ_N, respectively, being $\Gamma = \Gamma_D \cup \Gamma_N$, $\Gamma_D \cap \Gamma_N = \emptyset$ and $\Gamma_D \neq \emptyset$, for given $\bar{p}(\mathbf{x})$ and $\bar{q}(\mathbf{x})$:

$$p(\mathbf{x}) = \bar{p}(\mathbf{x}), \text{ on } \quad \Gamma_D, \tag{2}$$

$$q(\mathbf{x}) = \bar{q}(\mathbf{x}), \text{ on } \quad \Gamma_N, \tag{3}$$

where $q = \kappa(\partial p/\partial \mathbf{n})$ is the flux component along the unit vector $\mathbf{n}$ normal to Γ, oriented outward, and being κ a positive coefficient. More generally, Robin type boundary conditions can be imposed on Γ for a given $\bar{r}(\mathbf{x})$ function:

$$r(\mathbf{x}) = \kappa \frac{\partial p}{\partial \mathbf{n}}(\mathbf{x}) + ap(\mathbf{x}) = \bar{r}(\mathbf{x}), \qquad \text{on } \Gamma \tag{4}$$

where $a > 0$ is a coefficient. For the sake of simplicity, in what follows uniform boundary conditions are used and κ is set to 1.

The fundamental solution $p^\star(\mathbf{x}_k, \mathbf{x})$ of the Laplace operator in 2D is:

$$p^\star(\mathbf{x}_k, \mathbf{x}) = \frac{1}{2\pi} \ln \frac{1}{\rho(\mathbf{x}_k, \mathbf{x})}, \tag{5}$$

being $\mathbf{x}_k \in \bar{\Omega} = \Omega \cup \Gamma$ a collocation point and $\rho(\mathbf{x}_k, \mathbf{x})$ the distance between $\mathbf{x}_k$ and $\mathbf{x}$. Setting $q^\star(\mathbf{x}_k, \mathbf{x}) = \nabla p^\star \cdot \mathbf{n}$, by using Green's function and the divergence theorem, Equation (1) can be written in integral form as:

$$c_k p(\mathbf{x}_k) + \int_\Gamma q^\star(\mathbf{x}_k, \mathbf{x}) p(\mathbf{x}) \mathrm{d}\mathbf{x} = \int_\Gamma p^\star(\mathbf{x}_k, \mathbf{x}) q(\mathbf{x}) \mathrm{d}\mathbf{x} \tag{6}$$

where $\mathbf{x}_k$ is located on the boundary and c_k is a free term depending on the regularity of the boundary at $\mathbf{x}_k$; in particular, c_k is equal to ½ if Γ is smooth.

A mesh is introduced on the boundary Γ, along with an associated approximation of $p(\mathbf{x})$, $q(\mathbf{x})$, such that:

$$p = \sum_{j=1}^{N} p_j \varphi_j(\mathbf{x}), \qquad q = \sum_{j=1}^{N} q_j \varphi_j(\mathbf{x}),$$

with $p(\mathbf{x})$ and $q(\mathbf{x})$ now approximations in a finite dimensional space, p_j, q_j unknown nodal values and $\{\varphi_j\}_{j=1,\ldots,N}$ selected basis functions. By locating Equation (6) at each node $k = 1, \ldots, N$ of the discretization, the following set of equations follows:

$$c_k p_k + \sum_{j=1}^{N} H_{kj} p_j = \sum_{j=1}^{N} G_{kj} q_j \tag{7}$$

where H_{kj} and G_{kj} are the values of the integrals on Γ of the basis function φ_j against $q^\star$ and $p^\star$, respectively.

Equation (7) can be expressed in concise form as follows:

$$\mathbf{Hp} = \mathbf{Gq} \tag{8}$$

where $\mathbf{p}$ and $\mathbf{q}$ are the vectors of potential and flux nodal values, respectively, and $\mathbf{H}$ and $\mathbf{G}$ are the related influence coefficient matrices collecting coefficients H_{kj} and G_{kj} respectively, with diagonal terms H_{kk} also including coefficients c_k. In each node, one of two among p and q is known through a Dirichlet or Neumann condition, thus, columns of matrices $\mathbf{H}$ and $\mathbf{G}$ can be switched in order to have all the unknown coefficients on the left-hand side and all the known terms on the right-hand side. Finally, new matrices $\mathbf{A}$ and $\mathbf{B}$ and a new vector of unknown coefficients $\mathbf{X}$ are built and Equation (8) is re-arranged as follows:

$$\mathbf{AX} = \mathbf{Bb}, \tag{9}$$

where $\mathbf{b}$ is the column vector collecting all the known nodal values.

A Robin-type boundary condition on the whole Γ is set as follows:

$$q(\mathbf{x}) = \bar{r}(\mathbf{x}) - p(\mathbf{x}), \tag{10}$$

and Equation (7) becomes:

$$c_k p_k + \sum_{j=1}^{N} \left(H_{kj} - G_{kj}\right) p_j = R_k,$$

with:

$$R_k = \int_\Gamma p^\star(\mathbf{x}_k, \mathbf{x}) \bar{r}(\mathbf{x}) \mathrm{d}\mathbf{x}. \tag{11}$$

Given a domain Ω consisting of two sub-domains, such that $\bar{\Omega} = \bar{\Omega}_i \cup \bar{\Omega}_j$ (see for example domains Ω_i and Ω_j of Figure 2), with S the interface between the two domains, i.e. $S = \bar{\Omega}_i \cap \bar{\Omega}_j$, and given also $\Gamma_\zeta = \partial\Omega_\zeta \cap \Gamma$ border of Ω_ζ without interface S, for $\zeta = \{i, j\}$, Equation (9) can be written for Ω_i as follows[1]:

$$\begin{pmatrix} \mathbf{A}_{i,\Gamma\Gamma} & \mathbf{A}_{i,\Gamma S} \\ \mathbf{A}_{i,S\Gamma} & \mathbf{A}_{i,SS} \end{pmatrix} \begin{pmatrix} \mathbf{X}_{i,\Gamma} \\ \mathbf{X}_{i,S} \end{pmatrix} = \begin{pmatrix} \mathbf{B}_{i,\Gamma\Gamma} & \mathbf{B}_{i,\Gamma S} \\ \mathbf{B}_{i,S\Gamma} & \mathbf{B}_{i,SS} \end{pmatrix} \begin{pmatrix} \mathbf{b}_{i,\Gamma} \\ \mathbf{b}_{i,S} \end{pmatrix}, \tag{12}$$

where $\mathbf{X}_{i,\Gamma}$, $\mathbf{X}_{i,S}$ are the vectors of unknowns, associated to the nodes on Γ_i and S, respectively. Vectors $\mathbf{b}_{i,\Gamma}$, $\mathbf{b}_{i,S}$ collect boundary conditions terms, associated to discretization nodes on Γ_i and S, respectively, and

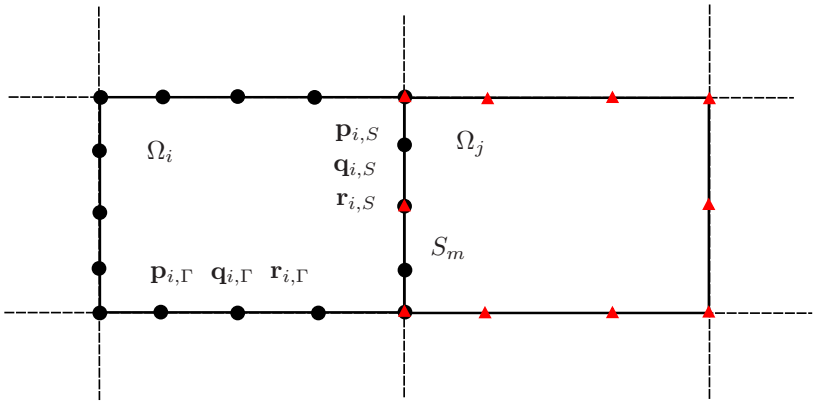


FIGURE 2 Typical BEM multiple domain problem: a common interface (S_m) is shared by two neighboring domains Ω_i and Ω_j; black dots denote the discretization of Ω_i and red triangles the one of Ω_j

matrices $\mathbf{A}_i$ and $\mathbf{B}_i$ are the re-arranged coefficient matrices, as discussed above. When Robin-type boundary conditions are enforced on the interface S, the equation system (12) becomes:

$$\begin{pmatrix} \mathbf{A}_{i,\Gamma\Gamma} & \mathbf{A}_{i,\Gamma S} + a\mathbf{B}_{i,\Gamma S} \\ \mathbf{A}_{i,S\Gamma} & \mathbf{A}_{i,SS} + a\mathbf{B}_{i,SS} \end{pmatrix} \begin{pmatrix} \mathbf{X}_{i,\Gamma} \\ \mathbf{p}_{i,S} \end{pmatrix} = \begin{pmatrix} \mathbf{B}_{i,\Gamma\Gamma} & \mathbf{B}_{i,\Gamma S} \\ \mathbf{B}_{i,S\Gamma} & \mathbf{B}_{i,SS} \end{pmatrix} \begin{pmatrix} \mathbf{b}_{i,\Gamma} \\ \mathbf{R}_{i,S} \end{pmatrix}, \tag{13}$$

in which $R_{i,S}$ is given as in Equation (11), for all nodes $\mathbf{x}_k$ on S.

3 | AIC ALGORITHMS

A domain Ω is considered, consisting of $\mathcal{I}$ non-overlapping connected sub-domains Ω_i (see Figure 2), such that $\bar{\Omega} = \bigcup_{i=1}^{\mathcal{I}} \bar{\Omega}_i$, and $\Omega_i \cap \Omega_j = \emptyset$, for all $i, j = 1, \dots, \mathcal{I}$, $i \neq j$; common interfaces between neighboring sub-domains are denoted by S_m, $m = 1, \dots, \mathcal{M}$, with $S_m = \bar{\Omega}_i \cap \bar{\Omega}_j$, being i, j the unique couple of indexes such that $\bar{\Omega}_i \cap \bar{\Omega}_j \neq \emptyset$. Multiple intersections between two sub-domains can also be considered, and are excluded here only for simplicity of notation. A function $\sigma : [1, \dots, \mathcal{M}] \mapsto [1, \dots, \mathcal{I}] \times [1, \dots, \mathcal{I}]$ is defined such that $\sigma(m) = \{i, j\}$ with $S_m = \bar{\Omega}_i \cap \bar{\Omega}_j$ and $i \neq j$. The boundary of Ω is Γ, with $\Gamma = \Gamma_{\mathrm{D}} \cup \Gamma_{\mathrm{N}}$, $\Gamma_{\mathrm{D}} \neq \emptyset$, $\Gamma_{\mathrm{N}} \cap \Gamma_{\mathrm{D}} = \emptyset$, whereas, for $i = 1, \dots, \mathcal{I}$, Γ_i is the portion of boundary of Ω_i that coincides with Γ, i.e. $\Gamma_i = \partial\Omega_i \cap \Gamma$. Functions p_i and q_i denote the potential and the flux, respectively, on the boundary of Ω_i, $i = 1, \dots, \mathcal{I}$, whereas p_{i,S_m} (resp. q_{i,S_m}) is the restriction of p_i (resp. q_i) to S_m for all $m \in \mathbb{M}_i$, being $\mathbb{M}_i$ the set of indexes $m = 1, \dots, \mathcal{M}$, such that $\partial\Omega_i \cap S_m \neq \emptyset$. Finally, for $i = 1, \dots, \mathcal{I}$, functions $p_{i,S_{\mathcal{M}}}$ are defined as:

$$p_{i,S_{\mathcal{M}}} = \left(p_{i,S_{m_1}}, p_{i,S_{m_2}}, \dots, p_{i,S_{m_{M_i}}} \right), \quad m \in \mathbb{M}_i,\ m_1 < m_2 < \cdots < m_{M_i}, \tag{14}$$

with M_i the cardinality of $\mathbb{M}_i$, and similarly for $q_{i,S_\mathcal{M}}$ and $r_{i,S_\mathcal{M}} = p_{i,S_\mathcal{M}} + aq_{i,S_\mathcal{M}}$.

In an IC algorithm, successive updates of the variables on the interfaces of the sub-domains are performed, until a given convergence criterion is fulfilled. The update of each interface variable p_{i,S_m}, q_{i,S_m}, $i = 1, \dots, \mathcal{I}$, depends on the values of the neighboring domain Ω_j, $\{i, j\} = \sigma(m)$. Thus, only small size linear systems need to be solved at each iteration[25]. As previously mentioned, a new AIC algorithm, P-RR, is proposed herein. Two AIC algorithms, namely parallel Dirichlet-Dirichlet and parallel Neumann-Neumann, and sequential Dirichlet-Neumann algorithm are considered in addition. In parallel algorithms, at each iteration, the solution for all the sub-domains is computed in parallel, rather than one sub-domain at a time as in the sequential algorithms. Dirichlet, Neumann or Robin denotes the type of boundary condition applied to each interface of two adjacent sub-domains. Obviously, for large multi-domain problems, the parallel algorithms result more appealing. Non conforming meshes are allowed at the interfaces, i.e. mesh elements on $\partial\Omega_i$ may not match the elements on $\partial\Omega_j$ on the common interface S_m, $\sigma(m) = \{i, j\}$ (see Figure 2 , where mesh element vertexes of $\partial\Omega_i$ are drawn as black dots and those of $\partial\Omega_j$ as red triangles). In these cases, the update of the interface variables requires a projection from one mesh to the other. This is a standard procedure, and in what follows, it is not explicitly mentioned in the description of the algorithms. It is worth to remark that these algorithms are not only suitable for BEM-BEM coupling problems, but can be easily extended to BEM-FEM or FEM-FEM coupling as well.

In the next sub-sections, the existing algorithms S-DN, P-DD, P-NN and the novel P-RR are described.

3.1 | Sequential Dirichlet-Neumann (S-DN) algorithm

The sequential Dirichlet-Neumann IC algorithm is presented here as term of comparison, given that it is widely used in BEM-BEM and BEM-FEM coupling problems and generally shows good convergence properties[25,26,27,28,29,30].

It is assumed that the geometry of the multi-domain problem is such that the indexes of domains Ω_i, $i = 1, \dots, \mathcal{I}$, can be divided into subsets $\mathcal{I}_\mathrm{D}$ and $\mathcal{I}_\mathrm{N}$, and each sub-domain Ω_d, $d \in \mathcal{I}_\mathrm{D}$, only shares interfaces with domains Ω_n with $n \in \mathcal{I}_\mathrm{N}$, or, equivalently, for each $m = 1, \dots, \mathcal{M}$, $\sigma(m) = \{d, n\}$ with $d \in \mathcal{I}_\mathrm{D}$ and $n \in \mathcal{I}_\mathrm{N}$. A sort of coloring criterion is applied, for instance, where a *black* or a *white* color is ascribed to each sub-domain, and only sub-domains with unmatched colors can be adjacent (thus sharing a segment interface). It is also required that $\partial\Omega_n \cap \Gamma_D \neq \emptyset$, for all $n \in \mathcal{I}_\mathrm{N}$. Clearly, this situation may not occur in

TABLE 2 Sequential Dirichlet-Neumann algorithm

1. set k=0 and initial guess $\mathbf{p}^0_{d,S_\mathcal{M}}$, $d \in \mathcal{I}_\mathrm{D}$
2. compute $\mathbf{q}^0_{d,S_m}$ by solving Equation (12) at Ω_d, $d \in \mathcal{I}_\mathrm{D}$, $m \in \mathcal{M}_d$
3. set initial guess $\mathbf{q}^0_{n,S_m} = -\mathbf{q}^0_{d,S_m}$ on each interface S_m, $m = 1, \dots, \mathcal{M}$ $(d, n) = \sigma(m)$
4. compute $\mathbf{p}^0_{n,S_m}$ $n \in \mathcal{I}_\mathrm{N}$ by solving Equation 12
5. DO
   (a) update $\mathbf{p}^{k+1}_{d,S_m} = \mathbf{p}^k_{d,S_m} - \alpha_{\mathrm{S-DN}}(\mathbf{p}^k_{d,S_m} - \mathbf{p}^k_{n,S_m})$ for each s.d. $d \in \mathcal{I}_\mathrm{D}$, $n \in \mathcal{M}_d$, $(d, n) = \sigma(m)$;
   (b) compute $\mathbf{q}^{k+1}_{d,S_m}$ by solving Equation (12) on each sub-domain Ω_d, $d \in \mathcal{I}_\mathrm{D}$
   (c) update $\mathbf{q}^{k+1}_{n,S_m} = -\mathbf{q}^{k+1}_{d,S_m}$ on each interface S_m, $m = 1, \dots, \mathcal{M}$, $(d, n) = \sigma(m)$
   (d) compute $\mathbf{p}^{k+1}_{n,S_m}$ by solving Equation (12) at Ω_N, $n \in \mathcal{I}_\mathrm{N}$
   (e) $k = k + 1$

   WHILE $\epsilon^\star = \sum_{i=1}^{\mathcal{I}} \|\mathbf{p}^{k+1}_{i,S_\mathcal{M}} - \mathbf{p}^k_{i,S_\mathcal{M}}\| / \|\mathbf{p}^k_{i,S_\mathcal{M}}\| <$ TOL.

large multi-domain problems; furthermore, the algorithm does not allow a parallel resolution, therefore the algorithm is not particularly suitable to the solution of large multi-domain problems.

The steps of the algorithm are summarized in Table 2 . A Dirichlet boundary conditions is first assumed on the interfaces of the sub-domains Ω_d, $d \in \mathcal{I}_\mathrm{D}$, and Equation (12) is solved to compute the interface fluxes $q_{d,S_\mathcal{M}}$, used in the subsequent step as Neumann boundary conditions $q_{n,S_\mathcal{M}} = -q_{d,S_\mathcal{M}}$ on the interfaces of sub-domains Ω_n, $n \in \mathcal{I}_\mathrm{N}$. Equation (12) is again solved on these interfaces to compute the potentials $p_{n,S_\mathcal{M}}$, used in turn to correct the initial assumption. At each iteration, the value of the relaxation parameter $\alpha_{\mathrm{S-DN}}$ is chosen as follows:

$$\alpha_{\mathrm{S-DN}} = \frac{\sum_{m=1}^{\mathcal{M}} (\boldsymbol{\epsilon}^p_{i,S_m}, \boldsymbol{\epsilon}^p_{i,S_m} - \boldsymbol{\epsilon}^p_{j,S_m})}{\sum_{m=1}^{\mathcal{M}} \|\boldsymbol{\epsilon}^p_{i,S_m} - \boldsymbol{\epsilon}^p_{j,S_m}\|^2}, \tag{15}$$

with $\{i, j\} = \sigma(m)$, and $\boldsymbol{\epsilon}^p_{i,S_m} = p^k_{i,S_m} - p^{k-1}_{i,S_m}$. Details on the derivation of the value of the relaxation parameter for the various algorithms are given in Section 3.5.

3.2 | Parallel Dirichlet-Dirichlet (P-DD) algorithm

The parallel Dirichlet-Dirichlet IC algorithm overcomes the previous limitations, by allowing the handling of multi-domain problems of any geometry and a parallel resolution of the problem on all the sub-domains at each iteration. Potential values are initially guessed for all the interfaces and updated according to the mismatch of the computed fluxes at the interfaces. The steps of P-DD are reported in Table 3 [24], and also

TABLE 3 Parallel Dirichlet-Dirichlet algorithm

1. set k=0 and initial guess $\mathbf{p}^0_{i,S_\mathcal{M}}$, for $i = 1, \dots, \mathcal{I}$
2. compute $\mathbf{q}^0_{i,S_\mathcal{M}}$, by solving Equation (12) at each sub-domain Ω_i, $i = 1, \dots, \mathcal{I}$
3. DO
   (a) update $\mathbf{p}^{k+1}_{i,S_m} = \mathbf{p}^k_{i,S_m} - \alpha_{\text{P-DD}}(\mathbf{q}^k_{i,S_m} + \mathbf{q}^k_{j,S_m})$, on each interface S_m, $m = 1, \dots, \mathcal{M}$, $\{i, j\} = \sigma(m)$
   (b) compute $\mathbf{q}^{k+1}_{i,S_\mathcal{M}}$ by solving Equation (12) at each sub-domain Ω_i, $i = 1, \dots, \mathcal{I}$
   (c) $k = k + 1$

   WHILE $\epsilon^\star = \sum_{i=1}^{\mathcal{I}} \|\mathbf{p}^{k+1}_{i,S_\mathcal{M}} - \mathbf{p}^k_{i,S_\mathcal{M}}\| / \|\mathbf{p}^k_{i,S_\mathcal{M}}\| <$ TOL.

TABLE 4 Parallel Neumann-Neumann algorithm

1. set k=0 and initial guess $\mathbf{q}^0_{i,S_\mathcal{M}}$ for $i = 1, \dots, \mathcal{I}$
2. compute $\mathbf{p}^0_{i,S_\mathcal{M}}$ solving Equation (12) on each sub-domain Ω_i, $i = 1, \dots, \mathcal{I}$
3. DO
   (a) update $\mathbf{q}^{k+1}_{i,S_m} = \mathbf{q}^k_{i,S_m} - \alpha_{\text{P-NN}}(\mathbf{p}^k_{i,S_m} - \mathbf{p}^k_{j,S_m})$ on each interface S_m, $m = 1, \dots, \mathcal{M}$, $\{i, j\} = \sigma(m)$
   (b) compute $\mathbf{p}^{k+1}_{i,S_\mathcal{M}}$ by solving Equation (12) at each sub-domain Ω_i, $i = 1, \dots, \mathcal{I}$
   (c) $k = k + 1$

   WHILE $\epsilon^\star = \sum_{i=1}^{\mathcal{I}} \|\mathbf{q}^{k+1}_{i,\mathcal{M}} - \mathbf{q}^k_{i,\mathcal{M}}\| / \|\mathbf{q}^k_{i,\mathcal{M}}\| <$ TOL.

in this case the relaxation parameter is dynamically computed at each iteration as (see Section 3.5):

$$\alpha_{\text{P-DD}} = \frac{\sum_{m=1}^{\mathcal{M}} (\boldsymbol{\epsilon}^p_{i,S_m}, \boldsymbol{\epsilon}^q_{i,S_m} + \boldsymbol{\epsilon}^q_{j,S_m})}{\sum_{m=1}^{\mathcal{M}} \|\boldsymbol{\epsilon}^q_{i,S_m} + \boldsymbol{\epsilon}^q_{j,S_m}\|^2}, \tag{16}$$

with $\{i, j\} = \sigma(m)$, and $\boldsymbol{\epsilon}^u_{i,S_m} = u^k_{i,S_m} - u^{k-1}_{i,S_m}$, being $u = p$ or $u = q$.

3.3 | Parallel Neumann-Neumann (P-NN) algorithm

Another commonly used IC algorithm is the parallel Neumann-Neumann (P-NN) algorithm in which Neumann boundary conditions are assumed and successively updated at the interfaces. However, if Neumann conditions are specified on the whole boundary of any sub-domain, the P-NN algorithm is not applicable, as this would lead to an ill-posed problem. The steps of P-NN are reported in Table 4 [25], and the relaxation parameter is dynamically computed as (see Section 3.5):

$$\alpha_{\text{P-NN}} = \frac{\sum_{m=1}^{\mathcal{M}} (\boldsymbol{\epsilon}^q_{i,S_m}, \boldsymbol{\epsilon}^p_{i,S_m} - \boldsymbol{\epsilon}^p_{j,S_m})}{\sum_{m=1}^{\mathcal{M}} \|\boldsymbol{\epsilon}^p_{i,S_m} - \boldsymbol{\epsilon}^p_{j,S_m}\|^2}, \tag{17}$$

TABLE 5 Parallel Robin-Robin (P-RR) algorithm

1. set k=0 and initial guess $\mathbf{r}^0_{i,S_\mathcal{M}}$ for $i = 1, \dots, \mathcal{I}$
2. compute $\mathbf{p}^0_{i,S_\mathcal{M}}$ and $\mathbf{q}^0_{i,S_\mathcal{M}}$ solving Equations (13) and (10) for each sub-domain Ω_i, $i = 1, \dots, \mathcal{I}$
3. DO
   (a) update $\mathbf{r}^{k+1}_{i,S_m} = \mathbf{r}^k_{i,S_m} - \alpha_{\text{P-RR}}[\mathbf{p}^k_{i,S_m} - \mathbf{p}^k_{j,S_m} - a(\mathbf{q}^k_{i,S_m} + \mathbf{q}^k_{j,S_m})]$ on each interface S_m, $m = 1, \dots, \mathcal{M}$, $\{i, j\} = \sigma(m)$
   (b) compute $\mathbf{p}^{k+1}_{i,S_\mathcal{M}}$ and $\mathbf{q}^{k+1}_{i,S_\mathcal{M}}$ solving Equations (13) and (10) for each sub-domain Ω_i, $i = 1, \dots, \mathcal{I}$
   (c) $k = k + 1$

   WHILE $\epsilon^\star = \sum_{i=1}^{\mathcal{I}} \|\mathbf{r}^{k+1}_{i,S_\mathcal{M}} - \mathbf{r}^k_{i,S_\mathcal{M}}\| / \|\mathbf{r}^k_{i,S_\mathcal{M}}\| <$ TOL.

with $\{i, j\} = \sigma(m)$, and $\epsilon^u_{i,S_m} = u^k_{i,S_m} - u^{k-1}_{i,S_m}$, being $u = p$ or $u = q$.

3.4 | Novel Parallel Robin-Robin (P-RR) algorithm

The new P-RR algorithm uses a Robin condition obtained as a linear combination of the P-DD and P-NN algorithm interface conditions; thus, this version of the P-RR algorithm differs from the one proposed by Lions[44], and later used for example by Elleithy and Tanaka[27]. P-RR algorithm overcomes the limitations of S-DN and P-NN and is an alternative to P-DD for solving large multi-domain problems. The steps of P-RR are reported in Table 5 . The relaxation parameter is chosen at each iteration as (see Section 3.5):

$$\alpha_{\text{P-RR}} = \frac{\sum_{m=1}^{\mathcal{M}} (\boldsymbol{\epsilon}^r_{i,S_m}, \boldsymbol{\epsilon}^p_{i,S_m} - \boldsymbol{\epsilon}^p_{j,S_m} + \boldsymbol{\epsilon}^q_{i,S_m} + \boldsymbol{\epsilon}^q_{j,S_m})}{\sum_{m=1}^{\mathcal{M}} \|\boldsymbol{\epsilon}^p_{i,S_m} - \boldsymbol{\epsilon}^p_{j,S_m} + \boldsymbol{\epsilon}^q_{i,S_m} + \boldsymbol{\epsilon}^q_{j,S_m}\|^2}, \tag{18}$$

with $\{i, j\} = \sigma(m)$, and $\epsilon^u_{i,S_m} = u^k_{i,S_m} - u^{k-1}_{i,S_m}$, being alternatively $u = p, q, r$.

It is known from literature that the convergence of IC algorithms highly depends on the value of α[25,26,27,28,29]. Generally, a constant, non-optimal value of α is used for all the iterations, with the expedient to select it small enough to ensure convergence[25,26,27]. The AIC algorithms here proposed are endowed with an adaptive relaxation parameter α, selected at each iteration, with the aim to speed-up the convergence. In the next sub-section, the derivation of α is reported for all the considered algorithms.

3.5 | Adaptive selection of relaxation parameter α

The following generalized potential $\tilde{\mathbf{p}}$, flux $\tilde{\mathbf{q}}$ and Robin term $\tilde{\mathbf{r}}$ are introduced as:

$$\begin{aligned}\tilde{\mathbf{p}}^k_{l,f,S_m} &= \mathbf{p}^k_{l,S_m},\\ \tilde{\mathbf{q}}^k_{l,f,S_m} &= (-1)^{(l\neq f)}\mathbf{q}^k_{l,S_m},\\ \tilde{\mathbf{r}}^k_{l,f,S_m} &= \mathbf{p}^k_{l,S_m} + a\tilde{\mathbf{q}}^k_{l,f,S_m}.\end{aligned} \tag{19}$$

For any interface S_m, $m = 1,\dots,\mathcal{M}$, the corrections, to be updated at each iteration, are re-written in terms of the above generalized quantities as:

$$\begin{aligned}\tilde{\mathbf{p}}^{k+1}_{i,i,S_m} &= \tilde{\mathbf{p}}^k_{i,i,S_m} - \alpha_{\text{S-DN}}(\tilde{\mathbf{p}}^k_{i,i,S_m} - \tilde{\mathbf{p}}^k_{j,i,S_m}),\\ \tilde{\mathbf{p}}^{k+1}_{i,i,S_m} &= \tilde{\mathbf{p}}^k_{i,i,S_m} - \alpha_{\text{P-DD}}(\tilde{\mathbf{q}}^k_{i,i,S_m} - \tilde{\mathbf{q}}^k_{j,i,S_m}),\\ \tilde{\mathbf{q}}^{k+1}_{i,i,S_m} &= \tilde{\mathbf{q}}^k_{i,i,S_m} - \alpha_{\text{P-NN}}(\tilde{\mathbf{p}}^k_{i,i,S_m} - \tilde{\mathbf{p}}^k_{j,i,S_m}),\\ \tilde{\mathbf{r}}^{k+1}_{i,i,S_m} &= \tilde{\mathbf{r}}^k_{i,i,S_m} - \alpha_{\text{P-RR}}(\tilde{\mathbf{r}}^k_{i,i,S_m} - \tilde{\mathbf{r}}^k_{j,i,S_m}),\end{aligned} \tag{20}$$

with $\{i,j\} = \sigma(m)$, or in compact form, as follows:

$$\tilde{\mathbf{v}}^{k+1}_{i,i,S_m} = \tilde{\mathbf{v}}^k_{i,i,S_m} - \alpha(\tilde{\mathbf{w}}^k_{i,i,S_m} - \tilde{\mathbf{w}}^k_{j,i,S_m}), \tag{21}$$

where $(\tilde{\mathbf{v}},\tilde{\mathbf{w}})$ may be one of the couples $(\mathbf{p},\mathbf{p})$, $(\mathbf{p},\mathbf{q})$, $(\mathbf{q},\mathbf{p})$, $(\mathbf{r},\mathbf{r})$, and again subscripts i, j refer to adjacent sub-domains Ω_i and Ω_j, respectively, with $\{i,j\} = \sigma(m)$ and α is one of the four relaxation parameters, according to the choice of $(\mathbf{v},\mathbf{w})$.

Correction terms are further defined as:

$$\epsilon^u_{l,f,S_m} = \mathbf{u}^k_{l,f,S_m} - \mathbf{u}^{k-1}_{l,f,S_m}, \tag{22}$$

with $\mathbf{u}$ equal to $\tilde{\mathbf{p}},\ \tilde{\mathbf{q}},\ \tilde{\mathbf{r}}$, giving:

$$\epsilon^{\tilde{p}}_{l,f,S_m} = \tilde{\mathbf{p}}^k_{l,f,S_m} - \tilde{\mathbf{p}}^{k-1}_{l,f,S_m}, \qquad \epsilon^{\tilde{q}}_{i,i,S_m} = \tilde{\mathbf{q}}^k_{l,f,S_m} - \tilde{\mathbf{q}}^{k-1}_{l,f,S_m}, \qquad \epsilon^{\tilde{r}}_{l,f,S_m} = \tilde{\mathbf{r}}^k_{l,f,S_m} - \tilde{\mathbf{r}}^{k-1}_{l,f,S_m}. \tag{23}$$

Following Lin et al.[28], a correction functional J is introduced, which equals to:

$$J(\alpha) = \sum_{m=1}^{\mathcal{M}} \|\epsilon^{\tilde{v}}_{i,i,S_m} - \alpha(\epsilon^{\tilde{w}}_{i,i,S_m} - \epsilon^{\tilde{w}}_{j,i,S_m})\|^2, \tag{24}$$

and the relaxation parameter α is computed as the minimizer of this functional at each iteration k, giving:

$$\alpha = \frac{\sum_{i=1}^{\mathcal{M}} (\boldsymbol{\epsilon}^{\tilde{v}}_{i,i,S_m}, \boldsymbol{\epsilon}^{\tilde{w}}_{i,i,S_m} - \boldsymbol{\epsilon}^{\tilde{w}}_{j,i,S_m})}{\sum_{i=1}^{\mathcal{M}} \|\boldsymbol{\epsilon}^{\tilde{w}}_{i,i,S_m} - \boldsymbol{\epsilon}^{\tilde{w}}_{j,i,S_m}\|^2} \tag{25}$$

in which again $(\mathbf{v}, \mathbf{w})$ may be one of the couples $(\mathbf{p}, \mathbf{p})$, $(\mathbf{p}, \mathbf{q})$, $(\mathbf{q}, \mathbf{p})$ or $(\mathbf{r}, \mathbf{r})$, corresponding to S-DN, P-DD, P-NN and P-RR, respectively.

Note that Soares[29] indicated that α should be constrained in the range $[0, 1]$. However, in successive works, Soares and Godinho[30] found that a non-constrained relaxation parameter may offer faster convergence. Thus, a non-constrained α for all the IC algorithm is used herein. The performances of the four proposed AIC algorithms are explored in the next Section, through the analysis of two multi-domain problems.

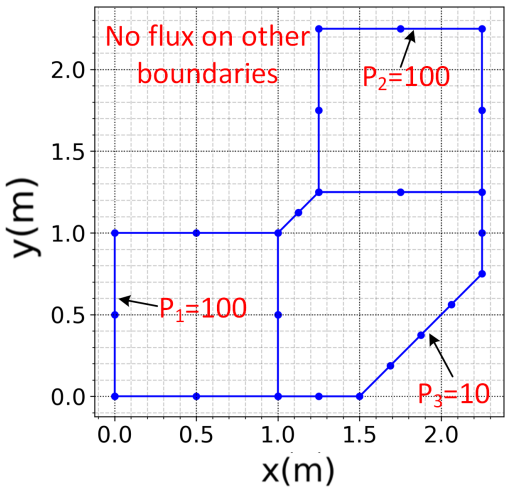
4 | NUMERICAL EXAMPLES

In this Section numerical results are presented to show applicability and effectiveness of the algorithms herein presented. In particular, the algorithms are first validated, and then the relative convergence rates are investigated in terms of number of iterations required to solve problems of increasing complexity until a prescribed convergence criterion is met. Also the sensitivity of the algorithms is analyzed with respect to several geometrical and discretization variables, such as the number of sub-domains, the mesh-size, the lack of mesh conformity at the interfaces, and the polynomial accuracy of BEM discretizations. Problems of increasing complexity are considered. The convergence criterion is $\epsilon^\star < \mathrm{TOL} = 10^{-5}$ for all the simulations. The parameter a in Equation (4) is set to one.

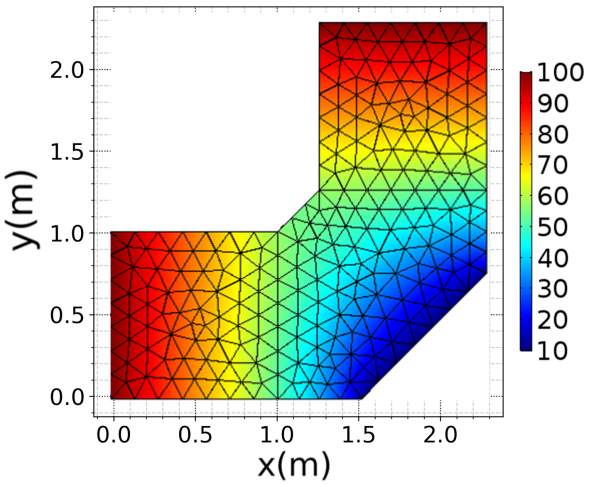
All the AIC algorithms are implemented in the Python code PyBEM2D.

4.1 | Test 1 L-Shape Problem

The first considered problem, labeled L-Shape problem, is set in the L-shaped domain (Figure 3 (a)), composed of three adjacent sub-domains. Potential values ($p_1 = p_2$=100) are prescribed at the left bottom edge and right top edge, respectively. An additional potential value (p_3=10) is prescribed at the right bottom edge. No-flow boundary conditions are prescribed on all other edges. As validation test, the solution obtained with the proposed algorithms is compared to a reference solution produced by the FEM code COMSOL[45]

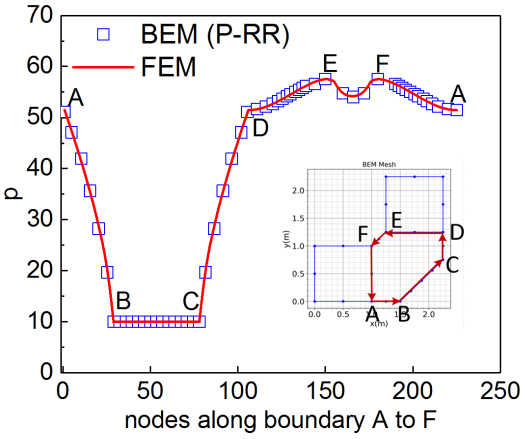


(a) 14-quadratic-elements (DOF=24) BEM discretization

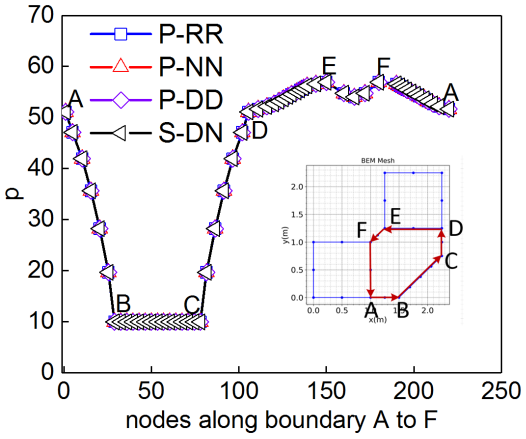


(b) 430-quadratic-elements (DOF=246) FEM discretization and solution *p*

FIGURE 3 L-Shape problem domain and reference solution



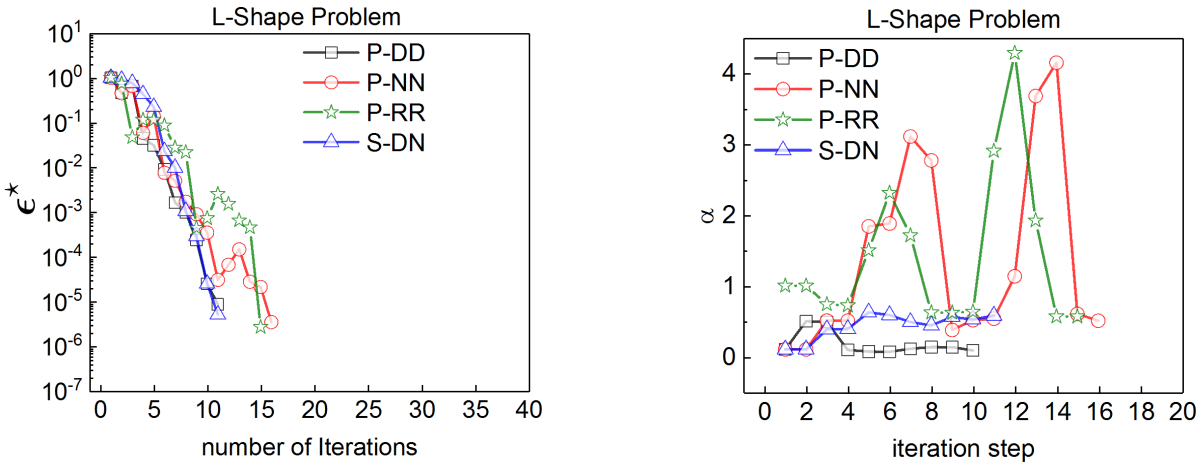
(a) BEM+P-RR vs. FEM



(b) All AIC algorithms

FIGURE 4 Comparison of potential solutions for the L-Shape problem

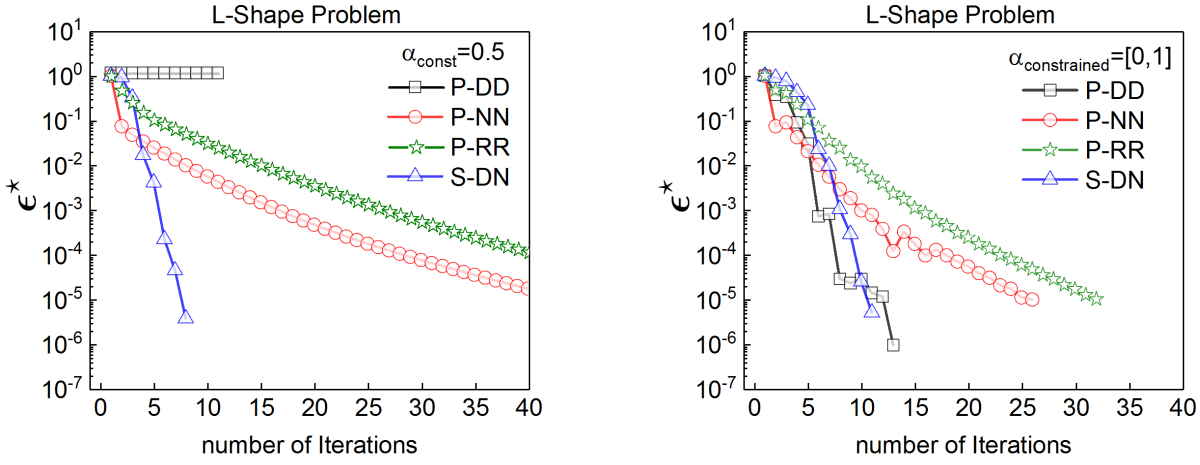
(Figure 3 (b)). Quadratic elements are used for both BEM and FEM. Figure 4 (a) shows a comparison, limited to the bottom-left portion of the domain, between the reference computed potential, in solid line, and the solution obtained by means of the P-RR algorithm on equally spaced nodes (not matching the computational mesh), tagged with square markers. It can be noticed that there is a perfect matching between the two solutions. With reference to the same sub-domain, in Figure 4 (b), the solutions of all the presented AIC algorithms are plotted, and again excellent agreements are shown.



(a) Relative corrections against iterations

(b) Values of α against iterations

FIGURE 5 Convergence history of the AIC algorithms for L-Shape problem

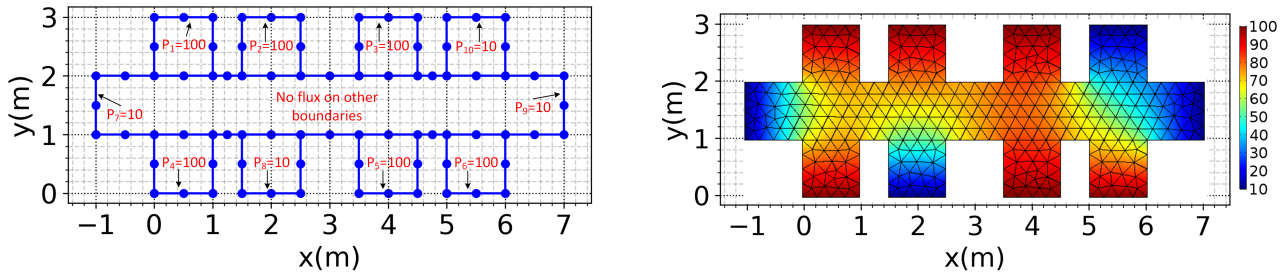


(a) Fixed α

(b) Constrained α

FIGURE 6 Convergence of AIC algorithms for fixed (left) and constrained (right) relaxation parameter

In Figure 5 (a), the relative correction $\epsilon^\star$ is plotted against the number of iterations with α chosen according to Equation (25). In Figure 5 (b) the evolution of α with iterations is shown. It can be stated that all the proposed algorithms require a number of iterations between 11 and 16 to reach the prescribed value of tolerance, and, as previously discussed, values of α outside the interval $[0, 1]$ are obtained throughout the iterations[24,25,26]. The advantage of the proposed adaptive selection of the relaxation parameter becomes more evident as shown in Figure 6 , where $\epsilon^\star$ is now plotted against the number of iterations when a fixed



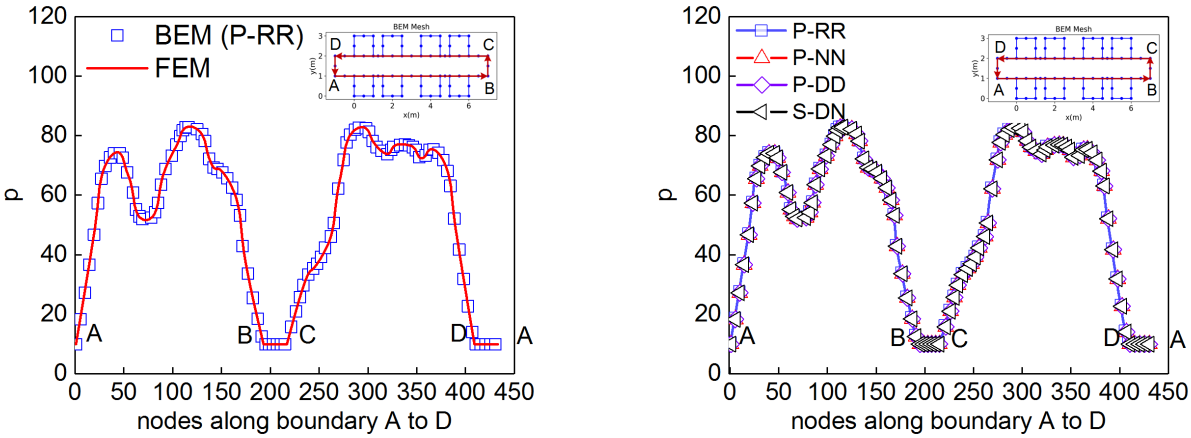
(a) 60-quadratic-elements (DOF=80) BEM discretization (b) 684-quadratic-elements (DOF=411) FEM discretization and solution p

FIGURE 7 Ship-Shape problem domain and reference solution

value $\alpha = \frac{1}{2}$ is used (Figure 6 (a)) or when α is constrained in the interval $[0, 1]$ (Figure 6 (b)): all the parallel AIC algorithms require a higher number of iterations in both these circumstances, and, when α is constant the P-DD algorithm does not converge, whereas P-NN and P-RR require more than 40 iterations to reach the stopping criterion. Only the S-DN algorithm appears to be almost unaffected by the choice of α, however, as mentioned before, this algorithm is not suitable for large multi-domain problems. These results show that the proposed adaptive selection of α allows the use of AIC algorithms in multi-domain problems without the need of a costly tuning for the definition of an optimal fixed value for α, that is usually a problem-dependent task[27]. Without the tuning, IC algorithms with fixed relaxation parameters could be even not usable, as convergence is not guaranteed.

4.2 | Test 2 Ship-Shape Problem

The second problem is defined on a multi-domain geometry composed of a central rectangle with eight squared domains symmetrically connected along the two longer edges, as shown in Figure 7 (a). This test is named Ship-Shape problem. Six high potential values (p_1 to p_6 equal to 100) are prescribed at the bottom and top edges. Four low potential values (p_7 to p_{10} equal to 10) are prescribed at the left, right, top and bottom edges of the central shaft, all other edges being insulated. A comparison with the reference solution obtained by using COMSOL (Figure 7 (b)) is proposed also in this case in Figure 8 , and again the excellent quality of the obtained solution with respect to the reference solution is shown. Results are reported in Figure 8 (a), for the comparison of the P-RR solution with the standard FEM solution, and in Figure 8 (b), where the results provided by all the AIC algorithms are compared. The evolution of $\epsilon^\star$ with respect to the number of iterations is reported in Figure 9 for all the considered AIC algorithms. It appears that the



(a) BEM+P-RR vs. FEM

(b) All AIC algorithms

FIGURE 8 Comparison of potential solutions for the Ship-Shape problem

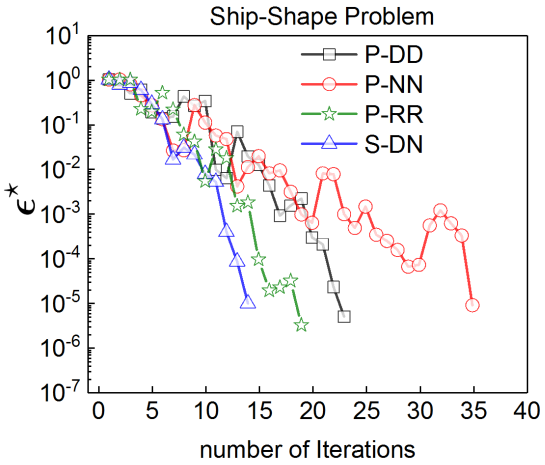


FIGURE 9 Convergence of the AIC algorithms with the number of iterations for the Ship-Shape problem

S-DN and the P-RR algorithms require only a slightly higher number of iterations to reach the prescribed tolerance, when compared to the simpler L-Shape problem, whereas P-DD and P-NN require about 65% and 130% more iterations, respectively, than the previous example.

4.3 | Sensitivity analysis

It is known that the optimal value for α, i.e. the value that minimizes the number of iterations to reach a given convergence criterion, depends on many problem-related variables, such as geometry, boundary

conditions, mesh-size, problem data[26,27], and static (non-adaptive) relaxation parameters are determined case by case by using a trial-and-error method.

In this note α is chosen according to Equation (25). A study concerning the effect of problem-dependent parameters on the response of the AIC algorithms is proposed. First, the sensitivity of the algorithms to mesh density, mismatching of the interface meshes and BEM element type is considered, with reference to the L-Shape problem, and then the response of the algorithms to an increase of the number of sub-domains is investigated. For this last analysis Ship-Shape-like problems with a variable number of sub-domains, ranging from 3 to 45, are considered. For all the proposed algorithms, Figure 10 (a) reports the number of iterations N_i required to reach the convergence criterion $\epsilon^\star < 10^{-5}$ when the number of discretization nodes is increased for the L-Shape domain; Figure 10 (b) displays the evolution of N_i when the mesh of the bottom-left sub-domain of the L-Shape problem is refined, keeping constant the mesh on the other sub-domains; in Figure 10 (c) N_i is plotted against the polynomial order of the BEM discretization on the L-Shape domain; finally, in Figure 10 (d), the evolution of N_i when the number of sub-domains increases is shown with reference to a modular Ship-Shape domain. As a general trend, it can be noticed that S-DN is almost insensitive to the variation of the quantities here considered, whereas the other approaches have different responses. On one hand, the P-DD, P-RR and P-NN algorithms appear equally affected by the mesh refinement, with N_i increasing as the number of nodes increases (Figure 10 (a)); on the other hand, P-DD and P-NN appear more influenced by the mesh non-conformity at the interfaces than P-RR (Figure 10 (b)), that seems to be almost unaffected. P-RR, P-DD and P-NN are only marginally affected by the polynomial order of the discretization (Figure 10 (c)), and only P-NN is consistently affected by the increase of the number of sub-domains (Figure 10 (d)), that causes in turn an increase in N_i. Again P-RR results almost unaffected. The robustness of the P-RR algorithm when dealing with complex problems also emerges by comparing the number of iterations required to meet the convergence criterion for the L-Shape problem (simpler) and for the Ship-Shape problem (more complex) in Figures 5 (a) and 9 , respectively.

To sum up, S-DN is independent of the mesh-related and geometrical features. As far as the parallel algorithms are concerned, similar convergence trends are experienced. However, only P-RR shows a high performance when dealing with large problems and non-conforming meshes at interfaces. This confirms that the P-RR algorithm can be an effective and efficient tool for the solution of large multi-domain potential problems.

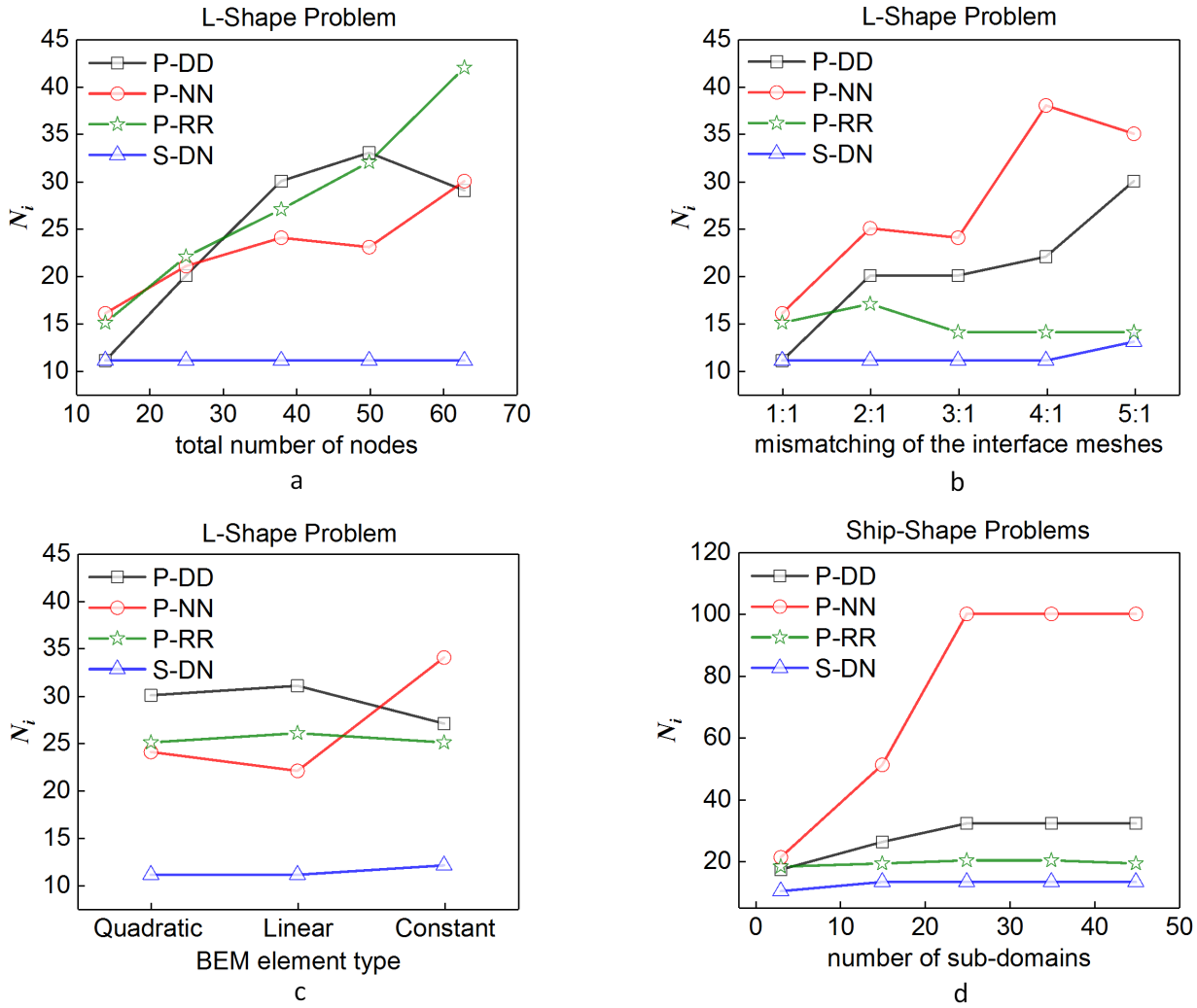


FIGURE 10 Number of iterations N_i plotted against: a) total number of nodes, (b) mismatching of the interface meshes, (c) BEM element type, (d) number of sub-domains

5 | CONCLUSION

In this note, the AIC algorithms S-DN, P-DD, P-NN for multi-domain problems solved by the Boundary Element Method (BEM) are revisited. The adaptive relaxation technique is generalized and extended to large multi-domain problems. A new parallel Robin-Robin (P-RR) AIC algorithm is also proposed. The convergence performance of S-DN, P-DD, P-NN and P-RR is investigated for comparison. The following conclusions can be drawn:

1. with reference to the tested examples, the P-RR algorithm is especially suitable for large multi-domain problems solved by using BEM and non-conforming meshes at the interfaces. In fact, in the two

test problems, results are almost unaffected by the increase of problem complexity (number of sub-domains) and by the use of non-matching meshes at the interfaces;

2. the sequential Dirichlet-Neumann (S-DN) algorithm shows a very good convergence performance, which can be seen from its fastest convergence in both the test problems, however its applicability to multi-domain problems is limited to very simple configurations;

3. the adaptive selection of α allows avoiding the tuning of a fixed α, so as to get rid of a costly and problem-dependent task.

ACKNOWLEDGEMENTS

This research was supported by Board of Regents of the State of Louisiana (LEQSF(2017-20)-RD-A-20). The authors express their appreciation to Dr. Boyun Guo (University of Louisiana at Lafayette) and Tianqi Ma (Fudan University) for their support. S.P. and S.S. acknowledge the support of INdAM-GNCS.

References

1. Brebbia, C.A. and Dominguez, J. 1994. Boundary elements: an introductory course. WIT press. Southampton. UK.

2. Katsikadelis, J.T., 2016. The Boundary Element Method for Engineers and Scientists: Theory and Applications. Academic Press. Oxford, UK

3. Wang, B., Du, J., Feng, Y., Wang, Y., Wang, S. and Yang, R., 2017, February. An Embedded Grid-Free Approach for Near Wellbore Streamline Simulation. 23rd SPE Reservoir Simulation Conference. Society of Petroleum Engineers. doi: https://doi.org/10.2118/182614-MS

4. Zhiming, C., Xinwei, L., Chenghui, H., Xiaoliang, Z., Langtao, Z., Yizhou, C., Heng, Y. and Zhenhua, C., 2014. Productivity estimations for vertically fractured wells with asymmetrical multiple fractures. Journal of Natural Gas Science and Engineering, 21, pp.1048-1060. doi: https://doi.org/10.1016/j.jngse.2014.10.025

5. Chen, Z., Liao, X., Zhao, X., Dou, X. and Zhu, L., 2015. Performance of horizontal wells with fracture networks in shale gas formation. Journal of Petroleum Science and Engineering, 133, pp.646-664. doi: https://doi.org/10.1016/j.petrol.2015.07.004

6. Wu, K. and Olson, J.E., 2015. A simplified three-dimensional displacement discontinuity method for multiple fracture simulations. RInternational Journal of Fracture, 193(2), pp.191-204. doi: https://doi.org/10.1007/s1070

7. Wu, K. and Olson, J.E., 2016. Numerical investigation of complex hydraulic-fracture development in naturally fractured reservoirs. SPE production & operations, 31(04), pp.300-309. doi: https://doi.org/10.2118/173326-PA

8. Shen, L. and Liu, Y.J., 2007. An adaptive fast multipole boundary element method for three-dimensional acoustic wave problems based on the Burton-Miller formulation. Computational Mechanics, 40(3), pp.461-472. doi: https:// doi.org/10.1007/s00466-006-0121-2

9. Hu, X., Wu, K., Song, X., Yu, W., Tang, J., Li, G., and Shen, Z. 2018. A New Model for Simulating Particle Transport in a Low-viscosity Fluid for Fluid-driven Fracturing. AIChE Journal. doi: https://doi.org/10.1002/aic.16183

10. Yang, K., Wang, J., Du, J.M., Peng, H.F. and Gao, X.W., 2017.Radial integration boundary element method for nonlinear heat conduction problems with temperature-dependent conductivity. International Journal of Heat and Mass Transfer, 104, pp. 1145-1151. doi: https://doi.org/10.1016/j.ijheatmasstransfer.2016.09.015.

11. Yang, K., Peng, H.F., Cui, M. and Gao, X.W., 2015. New analytical expressions in radial integration BEM for solving heat conduction problems with variable coefficients. Engineering Analysis with Boundary Elements, 50, pp.224-230. doi: https:// doi.org/10.1016/j.enganabound.2014.08.010.

12. Chen, Z., Liao, X., Zhao, X., Dou, X., Zhu, L. and Sanbo, L., 2017. A Finite-Conductivity Horizontal-Well Model for Pressure-Transient Analysis in Multiple-Fractured Horizontal Wells. SPE Journal. doi: https:// doi.org/10.2118/177230-PA

13. Chen, Z., Liao, X., Zhao, X., Lv, S. and Zhu, L., 2016. A semianalytical approach for obtaining type curves of multiple-fractured horizontal wells with secondary-fracture networks. SPE Journal, 21(02), pp.538-549. doi:https:// doi.org/10.2118/178913-PA

14. Lenti, V. and Fidelibus, C., 2003. A BEM solution of steady-state flow problems in discrete fracture networks with minimization of core storage. Computers & geosciences, 29(9), pp.1183-1190. doi: https://doi.org/10.1016/S0098-3004(03)00140-7

15. Liu, Y., 2009. Fast multipole boundary element method: theory and applications in engineering. Cambridge university press.

16. Benedetti, I., Aliabadi, M.H. and Davi, G., 2008. A fast 3D dual boundary element method based on hierarchical matrices. International journal of solids and structures, 45(7), pp.2355-2376. doi: https://doi.org/10.1016/j.ijsolstr.2007.11.018

17. Owens, J.D., Houston, M., Luebke, D., Green, S., Stone, J.E. and Phillips, J.C., 2008. GPU computing. Proceedings of the IEEE, 96(5), pp.879-899. doi: https://doi.org/10.1109/JPROC.2008.917757

18. Takahashi, T. and Hamada, T., 2009. GPU-accelerated boundary element method for Helmholtz'equation in three dimensions. International Journal for Numerical Methods in Engineering, 80(10), pp.1295-1321. doi: https://doi.org/ 10.1002/nme.2661

19. Cooper, C.D., Bardhan, J.P. and Barba, L.A., 2014. A biomolecular electrostatics solver using Python, GPUs and boundary elements that can handle solvent-filled cavities and Stern layers. Computer physics communications, 185(3), pp.720-729. doi: doi: https://doi.org/10.1016/j.cpc.2013.10.028

20. Torky, A.A. and Rashed, Y.F., 2017. GPU acceleration of the boundary element method for shear-deformable bending of plates. Engineering Analysis with Boundary Elements, 74, pp.34-48. doi: https://doi.org/10.1016/j.enganabound.2016.10.006

21. Tao, S., Cheng, J. and Mosallaei, H., 2016. An integral equation based domain decomposition method for solving large-size substrate-supported aperiodic plasmonic array platforms. MRS Communications, 6(2), pp.105-115. doi: https://doi.org/10.1557/mrc.2016.11

22. M. C. Cacas, M. C., Ledoux, E., de Marsily, G., Tillie, B., Barbreau, A., Durand, E., Feuga, B., and Peaudecerf, P., 1990. Modeling fracture flow with a stochastic discrete fracture network: calibration and validation: 1. The flow model. Water Resour. Res., 26, pp. 479-489. doi: http://dx.doi.org/10.1029/WR026i003p00479

23. Fidelibus, C., Cammarata, G., and Cravero, M., 2009. Hydraulic characterization of fractured rocks. In: Abbie M, Bedford JS (eds) Rock mechanics: new research. Nova Science Publishers Inc., New York.

24. Gao, X.W., Guo, L. and Zhang, C., 2007. Three-step multi-domain BEM solver for nonhomogeneous material problems. Engineering Analysis with Boundary Elements, 31(12), pp.965-973. doi: https://doi.org/10.1016/j.enganabound.2007.06.002

25. Kamiya, N., Iwase, H. and Kita, E., 1996. Parallel implementation of boundary element method with domain decomposition. Engineering Analysis with Boundary Elements, 18(3), 209-216. doi: 10.1016/S0955- 7997(96)00050-1

26. Elleithy, W.M., Al-Gahtani, H.J. and El-Gebeily, M., 2001. Iterative coupling of BE and FE methods in elastostatics. Engineering Analysis with Boundary Elements, 25(8), 685-695. doi: 10.1016/S0955-7997(01)00054-6

27. Elleithy, W.M. and Tanaka, M., 2003. Interface relaxation algorithms for BEM-BEM coupling and FEM-BEM coupling. Computer Methods in Applied Mechanics and Engineering, 192(26), pp.2977-2992. doi: 10.1016/S0045-7825(03)00312-8

28. Lin, C.C., Lawton, E.C., Caliendo, J.A. and Anderson, L.R., 1996. An iterative finite element-boundary element algorithm. Computers & Structures, 59(5), pp.899-909. doi: https://doi.org/10.1016/0045-7949(95)00285-5

29. Soares, D., 2008. An optimised FEM-BEM time-domain iterative coupling algorithm for dynamic analyses. Computers & Structures, 86(19), pp.1839-1844.

30. Soares, D. and Godinho, L., 2016. Heat conduction analysis by adaptive iterative BEM-FEM coupling procedures. Engineering Analysis with Boundary Elements, 73, pp.79-94.

31. Francois, S., Coulier, P. and Degrande, G., 2015. Finite element-boundary element coupling algorithms for transient elastodynamics. Engineering Analysis with Boundary Elements, 55, pp.104-121. doi: https://doi.org/10.1016/j.enganabound.2014.11.028

32. Bendali, Abderrahmane, and Yassine Boubendir. 2006. Non-overlapping domain decomposition method for a nodal finite element method. Numerische Mathematik 103.4, 515-537. doi: https://doi.org/10.1007/s00211-006-0010-9

33. Bendali, A., Boubendir, Y., & Fares, M. 2007. A FETI-like domain decomposition method for coupling finite elements and boundary elements in large-size problems of acoustic scattering. Computers & structures, 85(9), 526-535. doi: https://doi.org/10.1016/j.compstruc.2006.08.029

34. Boubendir, Y. 2007. An analysis of the BEM-FEM non-overlapping domain decomposition method for a scattering problem. Journal of computational and applied mathematics, 204(2), 282-291. doi: https://doi.org/10.1016/j.cam.2006.02.044

35. Boubendir, Y., Antoine, X., & Geuzaine, C. 2012. A quasi-optimal non-overlapping domain decomposition algorithm for the Helmholtz equation. Journal of Computational Physics, 231(2), 262-280. doi: https://doi.org/10.1016/j.jcp.2011.08.007

36. Cowsar, L.C., Mandel, J. and Wheeler, M.F., 1995. Balancing domain decomposition for mixed finite elements. Mathematics of computation, 64(211), pp.989-1015. doi: 10.1090/S0025-5718-1995-1297465-9 https://doi.org/10.1090/S0025-5718-1995-1297465-9

37. Farhat, C. and Roux, F.X., 1991. A method of finite element tearing and interconnecting and its parallel solution algorithm. International Journal for Numerical Methods in Engineering, 32(6), pp.1205-1227. doi: https://doi.org/10.1002/nme.16 20320604

38. Berrone, S., Pieraccini, S. and Scialò, S., 2013. A PDE-constrained optimization formulation for discrete fracture network flows. SIAM Journal on Scientific Computing, 35(2), pp.B487-B510. doi: https://doi.org/10.1137/120865884

39. Berrone, S., Pieraccini, S. and Scialò, S., 2014. An optimization approach for large scale simulations of discrete fracture network flows. Journal of Computational Physics, 256, pp.838-853. doi: https://doi.org/10.1016/j.jcp.2013.09.028

40. Berrone, S., Fidelibus, C., Pieraccini, S. and Scialò, S., 2014. Simulation of the steady-state flow in discrete fracture networks with non-conforming meshes and extended finite elements. Rock mechanics and rock engineering, 47(6), pp.2171-2182. doi: https://doi.org/10.1007/s00603-013-0513-5

41. Berrone, S., Pieraccini, S., Scialò, S. and Vicini, F., 2015. A parallel solver for large scale DFN flow simulations. SIAM Journal on Scientific Computing, 37(3), pp. C285-C306. doi: https://doi.org/10.1137/140984014

42. Berrone, S., Borio, A. and Scialò, S., 2016. A posteriori error estimate for a PDE-constrained optimization formulation for the flow in DFNs. SIAM Journal on Numerical Analysis, 54(1), pp. 242-261. doi: https://doi.org/10.1137/15M1014760

43. Berrone, S., Pieraccini, S. and Scialò, S., 2017. Flow simulations in porous media with immersed intersecting fractures. Journal of Computational Physics, 345, pp. 768–791. doi: https://doi.org/10.1016/j.jcp.2017.05.049

44. P.L. Lions, 1990. On the Schwarz alternating method: A variant for nonoverlapping subdomains. In R. Glowinsky, G.H. Golub, G.A. Meurant, J. Periaud (Eds.), Domain Decomposition Methods for Partial Differential Equations, SIAM, pp. 202-223

45. COMSOL, 2016. Multiphysics Reference Guide for COMSOL 5.2

How to cite this article: B. Wang, Y. Feng, S. Pieraccini, S. Scialò, and C. Fidelibus (2018), Iterative coupling of boundary element method with domain decomposition, *IJNME*, *2018;XX:XX–XX*.